z/OS

Resource Measurement Facility Programmer's Guide

Version 1 Release 13

SC33-7994-14

z/OS

Resource Measurement Facility Programmer's Guide

Version 1 Release 13

SC33-7994-14

Note

Before using this information and the product it supports, be sure to read the general information under "Notices" on page 341.

This edition applies to Version 1 Release 13 of z/OS (5694-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC33-7994–13.

Contents

Figures

Tables

About this document

The Resource Measurement Facility (RMF™) is a performance management tool that measures selected areas of system activity and presents the data collected in the form of System Management Facility (SMF) records, formatted printed reports, or formatted display reports. You can use this data to evaluate system performance and identify reasons for performance problems.

This document contains information and reference material to enable you to use RMF data for application programming. There is a number of different ways of getting at different kinds of information, and each one is described in a separate chapter of this document.

Further processing of RMF report data can also be done using spreadsheets. The Spreadsheet Reporter is described in the *z/OS RMF User's Guide*.

In addition, this document is describing diagnosis procedures that can be used in case of an error when running RMF.

Who should use this document

This document is intended for use by system programmers responsible for the development of individual, installation-specific applications in the area of system measurement. Because RMF is a tool for measuring MVS™ system performance, this document assumes that the reader has extensive knowledge of the MVS system.

For an overview of RMF, and guidance on using the standard capabilities of the product, see the *z/OS RMF User's Guide*.

How this document is organized

This document contains the following chapters:

Chapter 1, "SMF records"
: These are the records from which RMF obtains information for the standard reports. You can find all the information you need to use them for your own reports in this chapter.

Chapter 2, "RMF sysplex data services"
: These are callable services with which you as an RMF user can access performance data sysplex-wide. The calls, return codes and data layouts are described here.

Chapter 3, "Accessing performance data using the RMF Distributed Data Server"
: Application programs which want to retrieve sysplex-wide performance data can use the HTTP API of the Distributed Data Server (DDS). The DDS gathers data from the RMF instances running on the sysplex members. An application program can send an HTTP request for selected performance metrics to the DDS. The DDS returns the requested RMF data as a structured XML document. Thus, exploiters of the DDS HTTP API have instant access to a great variety of z/OS performance metrics including short-term information as well as long-term historical data.

RMF also provides CIM-based performance data gatherers for AIX®, Linux on System x®, and Linux on System z®. Therefore, exploiters of the DDS HTTP API can send an HTTP request to retrieve performance data from the endpoints running the supported Linux or AIX operating systems.

This topic describes how to specify HTTP requests to the DDS for performance data and how to interpret an XML document returned by the DDS.

Chapter 4, "z/OS CIM monitoring"

With z/OS V1.7 base element **Common Information Model (CIM)**, it is possible to use the DMTF CIM open standard for systems management. z/OS CIM implements the CIM server which is based on the OpenPegasus open source project. A CIM monitoring client invokes the CIM server which, in turn, collects z/OS metrics from the system and returns it to the calling client. To get the z/OS metrics, the CIM server invokes the z/OS RMF monitoring provider which retrieves the metrics associated with z/OS system resources. The z/OS RMF monitoring provider uses existing and extended RMF Monitor III performance data.

The z/OS metrics obtained by z/OS **CIM** are described in this topic. They are common across eServer platforms, so you can use them to create end-to-end monitoring applications.

Chapter 5, "Adding Monitor I and Monitor II installation exits"

You can enhance the gathering capabilities of Monitor I and add your own report types to Monitor II by writing your own exit routines. Details on coding and installing these exit routines are given in this chapter.

Chapter 6, "Adding Monitor III user exits"

The RMF Monitor III Utility helps you to add your own processing to the standard Monitor III reporting. This chapter describes this utility and its usage.

Chapter 7, "Using Monitor III VSAM data set support"

The processing and format of the VSAM data sets that Monitor III uses to store its information are described in this chapter.

Chapter 8, "Monitor III data reporter tables"

When coding Monitor III exit routines, for example, with the help of the Monitor III Utility, you have to know what information RMF has stored where for use in which reports. The data is stored in tables, and the layouts of these are shown here.

Chapter 9, "Diagnosing problems in RMF"

In this chapter, you find procedures that you might use in case of an error when running RMF.

The z/OS RMF library

This table shows the full titles and order numbers of the books in the RMF library for z/OS.

Table 1. RMF Library

Title	Order Number
z/OS RMF User's Guide	SC33-7990
z/OS RMF Report Analysis	SC33-7991
z/OS RMF Performance Management Guide	SC33-7992
z/OS RMF Messages and Codes	SC33-7993
z/OS RMF Programmer's Guide	SC33-7994
z/OS RMF Reference Summary	SX33-9033
Softcopy documentation as part of the *z/OS Collection* (SK3T-4269 (CD-Rom) and SK3T-4271 (DVD))	

How to read syntax diagrams

This section describes how to read syntax diagrams. It defines syntax diagram symbols, items that may be contained within the diagrams (keywords, variables, delimiters, operators, fragment references, operands) and provides syntax examples that contain these items.

Syntax diagrams pictorially display the order and parts (options and arguments) that comprise a command statement. They are read from left to right and from top to bottom, following the main path of the horizontal line.

For users accessing the Information Center using a screen reader, syntax diagrams are provided in dotted decimal format.

Symbols

The following symbols may be displayed in syntax diagrams:

Symbol	Definition
►►──	Indicates the beginning of the syntax diagram.
──►	Indicates that the syntax diagram is continued to the next line.
►──	Indicates that the syntax is continued from the previous line.
──►◄	Indicates the end of the syntax diagram.

Syntax items

Syntax diagrams contain many different items. Syntax items include:

- Keywords - a command name or any other literal information.
- Variables - variables are italicized, appear in lowercase, and represent the name of values you can supply.
- Delimiters - delimiters indicate the start or end of keywords, variables, or operators. For example, a left parenthesis is a delimiter.
- Operators - operators include add (+), subtract (-), multiply (*), divide (/), equal (=), and other mathematical operations that may need to be performed.

- Fragment references - a part of a syntax diagram, separated from the diagram to show greater detail.
- Separators - a separator separates keywords, variables or operators. For example, a comma (,) is a separator.

Note: If a syntax diagram shows a character that is not alphanumeric (for example, parentheses, periods, commas, equal signs, a blank space), enter the character as part of the syntax.

Keywords, variables, and operators may be displayed as required, optional, or default. Fragments, separators, and delimiters may be displayed as required or optional.

Item type	Definition
Required	Required items are displayed on the main path of the horizontal line.
Optional	Optional items are displayed below the main path of the horizontal line.
Default	Default items are displayed above the main path of the horizontal line.

Syntax examples

The following table provides syntax examples.

Table 2. Syntax examples

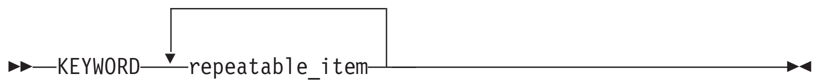
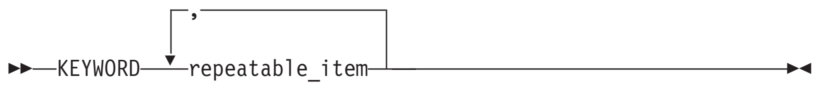
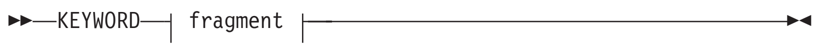
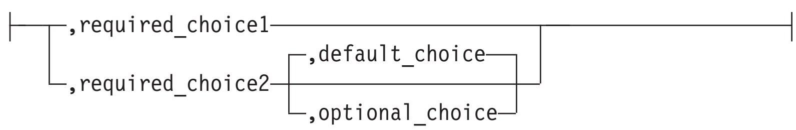
Item	Syntax example
Required item. Required items appear on the main path of the horizontal line. You must specify these items.	`►►──KEYWORD──required_item──────────►◄`
Required choice. A required choice (two or more items) appears in a vertical stack on the main path of the horizontal line. You must choose one of the items in the stack.	`►►──KEYWORD──┬─required_choice1─┬────►◄` `             └─required_choice2─┘`
Optional item. Optional items appear below the main path of the horizontal line.	`►►──KEYWORD──┬───────────────┬───────►◄` `             └─optional_item─┘`

Optional choice.
 	
An optional choice (two or more items) appears in a vertical stack below the main path of the horizontal line. You may choose one of the items in the stack.
	`►►──KEYWORD──┬──────────────────┬────►◄`
	`             ├─optional_choice1─┤`
	`             └─optional_choice2─┘`

Default.
 	
Default items appear above the main path of the horizontal line. The remaining items (required or optional) appear on (required) or below (optional) the main path of the horizontal line. The following example displays a default with optional items.
	`             ┌─default_choice1──┐`
	`►►──KEYWORD──┼──────────────────┼────►◄`
	`             ├─optional_choice2─┤`
	`             └─optional_choice3─┘`

Variable. Variables appear in lowercase italics. They represent names or values.	`►►──KEYWORD──`*`variable`*`─────────────►◄`

Table 2. Syntax examples (continued)

Item	Syntax example
Repeatable item. An arrow returning to the left above the main path of the horizontal line indicates an item that can be repeated. A character within the arrow means you must separate repeated items with that character. An arrow returning to the left above a group of repeatable items indicates that one of the items can be selected,or a single item can be repeated.	 
Fragment. The fragment symbol indicates that a labelled group is described below the main syntax diagram. Syntax is occasionally broken into fragments if the inclusion of the fragment would overly complicate the main syntax diagram.	 **fragment:** 

How to send your comments to IBM

We appreciate your input on this publication. Feel free to comment on the clarity, accuracy, and completeness of the information or give us any other feedback that you might have.

Use one of the following methods to send us your comments:

1. Send an email to s390id@de.ibm.com
2. Visit the RMF home page at `http://www.ibm.com/systems/z/os/zos/features/rmf/`
3. Visit the Contact z/OS web page at `http://www.ibm.com/systems/z/os/zos/webqs.html`
4. Mail the comments to the following address:
   IBM Deutschland Research & Development GmbH
   Department 3248
   Schönaicher Str. 220
   D-71032 Böblingen
   Federal Republic of Germany
5. Fax the comments to us as follows:
   From Germany: 07031-16-3456
   From all other countries: +(49)-7031-16-3456

Include the following information:

- Your name and address
- Your email address
- Your telephone or fax number
- The publication title and order number:
  z/OS V1R13.0 RMF Programmer's Guide
  SC33-7994-14
- The topic and page number related to your comment
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you submit.

If you have a technical problem

Do not use the feedback methods listed above. Instead, do one of the following:

- Contact your IBM service representative
- Call IBM technical support
- Visit the IBM zSeries support web page at `http://www.ibm.com/systems/z/support/`.

Summary of changes

This document contains information previously presented in *z/OS RMF Programmer's Guide*, SC33-7994-13.

This document includes terminology, maintenance, and editorial changes. The following information describes the enhancements that are being distributed with z/OS Version 1 Release 13. All technical changes or additions to the text are indicated by a vertical line to the left of the change.

What's new in z/OS V1R13

Resource monitoring of systems running Linux or AIX

RMF provides new CIM-based performance data gatherers for AIX on System p, Linux on System x, and Linux on System z. Exploiters of the HTTP API provided by the DDS can send an HTTP request to retrieve performance data from the endpoints running the Linux or AIX operating systems. The DDS returns the requested data as a structured XML document. With the Resource Monitoring plug-in for ***IBM z/OS Management Facility (z/OSMF)***, performance metrics from connected AIX or Linux systems can be displayed in the same way and together with z/OS in heterogeneous customer environments.

Monitor III VSAM data set support tables

- one small change in ERBASIG3

History of changes

What's new in z/OS V1R12

DDS API provides access to historical Postprocessor data

The Distributed Data Server (DDS) is enhanced to grant access to a selection of long-term historical Postprocessor data available with the following reports:

- CPU, DEVICE, CRYPTO, ESS, FCD, and OMVS (single system reports)
- WLMGL (sysplex report)
- OVERVIEW report

Application programs can send URL requests for historical RMF data to the DDS. Similar to the RMF Monitor III performance data, the requested Postprocessor data is exported to the requestor in a documented XML format for further processing.

Monitor III VSAM data set support tables

- ERBCPCDB

Monitor III data reporter tables

- ERBCPCT3

Also, fields available in the header of the Monitor III SYSINFO report are documented.

What's new in z/OS V1R11

For z/OS V1R11, there are two editions of this document, SC33-7994-11 and SC33-7994-12.

What's new in SC33-7994-12

Documentation of the DDS API: The Distributed Data Server (DDS) grants direct access to Monitor III RMF performance data. The DDS API is now published in the current edition of this document. By exploiting this API, customers can write their own performance management application programs.

What's new in SC33-7994-11

This edition describes measurements in the following tables:

Monitor III data reporter tables:

- ERBCHAT3
- ERBCPCT3

Also, fields available in the header of the Monitor III CPC report are documented.

Monitor III VSAM data set support tables:

- ERBCPCDB

What's new in z/OS V1R10

64-Bit Common Storage measurements

RMF provides measurements of 64-bit common storage consumption in the following Monitor III data gatherer tables:

- ERBASIG3
- ERBGEIG3

Also, Monitor III data reporter table ERBSTMT3 is introduced for the new Monitor III ***Storage Memory Objects*** report.

New reports on spin locks and suspend locks

Two new tables are introduced to support lock reporting:

- Table ERBLSPT3 is available for the new **Spin Lock Report** which provides information about spin lock holders and about jobs that are spinning because of a lock request.
- Table ERBLSUT3 is available for the new **Suspend Lock Report** which provides information about jobs that hold a suspend lock and about jobs that are suspended.

LDAP support removed

Starting with z/OS V1R10, you can no longer access RMF performance data via the LDAP interface. Instead, you can use the Common Information Model (CIM) monitoring interface, which has been provided since z/OS V1R7.

What's new in z/OS V1R9

For z/OS V1R9, there are two updates of this document, SC33-7994-09 and SC33-7994-08.

What's new in SC33-7994-09

Monitoring the usage of memory objects: RMF provides new metrics of the usage of memory objects by enhancing the following tables:

Monitor III VSAM data set support
- ERBASIG3
- ERBGEIG3

Monitor III data reporter tables
- ERBSTFT3

What's new in SC33-7994-08

RMF CIM based monitoring: RMF provides new metrics to monitor processor utilization (including zIIPs and zAAPs), to measure disk activity and to retrieve information about the active WLM policy.

Support of WLM's blocked workload promotion: If the CPU utilization of a system is at 100%, workloads with low importance (low dispatch priority) might not get dispatched anymore. This could cause problems if the work holds a resource and by that holds up more important workloads. Therefore, any address space or enclave which has ready-to-run work units but does not get CPU service within a certain time interval due to its low dispatch priority, will be temporarily promoted to a higher dispatch priority.

RMF provides new metrics of the handling of promoted work units by enhancing the Monitor III resource collection table ERBRCDG3.

HyperPAV support: The Monitor III device table ERBDVTG3 is extended to include new metrics for the support of HyperPAV devices.

New coupling facility measurements: RMF provides new measurements on CF processor utilization in the following Monitor III data reporter tables:
- ERBCFAT3
- ERBCFOT3
- ERBCFST3

What's new in z/OS V1R8

RMF CIM based monitoring

For z/OS 1.8, support for indications was added to the Common Information Model (CIM) infrastructure. RMF can now generate indications for monitoring data, this way enabling CIM clients to support event-based monitoring. A CIM client can subscribe for certain conditions, for example whether a certain performance metric value is above a threshold. While the subscription is active, the RMF indication provider checks the condition independently and delivers a CIM indication to notify the subscribed CIM client that the condition became true.

Support of IBM System z9 Integrated Information Processors (zIIP)

RMF provides measurements for IBM System z9 Integrated Information Processor (zIIP) activity by enhancing the following tables:

Monitor III VSAM data set support

- ERBASIG3
- ERBCPUG3
- ERBENCG33
- ERBRCDG3
- ERBUWDG3

Monitor III data reporter tables
- ERBENCT3
- ERBPRCT3
- ERBSYST3
- ERBPRUT3 (new)

Terminology:
RMF provides a consistent naming for zAAPs and zIIPs. The term IFA which was used for the zAAP in all RMF reports, help panels and manuals is now replaced by AAP. That is, AAP is used for the zAAP and IIP is used for the zIIP as field names.

What's new in z/OS V1R7

Monitor III Data Reporter Tables

Two new tables are introduced to support monitoring of zFS UNIX file system activity:
- Table ERBZFST3 is available for the new Monitor III ***zFS Summary Report*** which helps to control the zFS environment and delivers information on which tuning possibilities of zFS could be applied.
- Table ERBZFAT3 is available for the new Monitor III ***zFS Activity Report*** which measures zFS activity on the basis of single file systems, for example, a file system's performance and DASD utilization.

Two new tables are introduced to support disk space monitoring:
- Table ERBSPDT3 is available for the new Monitor III ***Disk Space Report*** which provides information about capacity and available space for defined storage groups.
- Table ERBSPGT3 is available for the new Monitor III ***Storage Space Report*** which provides capacity and space information for volumes belonging to the defined storage groups.

RMF CIM based monitoring

With z/OS V1R7 base element **Common Information Model (CIM)**, it is possible to use the DMTF CIM open standard for systems management. z/OS CIM implements the CIM server which is based on the OpenPegasus open source project. A CIM monitoring client invokes the CIM server which, in turn, collects z/OS metrics from the system and returns it to the calling client. To get the z/OS metrics, the CIM server invokes the z/OS RMF monitoring provider which retrieves the metrics associated with z/OS system resources. The z/OS RMF monitoring provider uses existing and extended RMF Monitor III performance data. The metrics obtained by this new API are common across eServer platforms, so you can use it to create end-to-end monitoring applications.

What's new in z/OS V1R6

Support of the eServer zSeries® Application Assist Processor

z/OS V1R6 provides the ability to run Java applications on a new type of processor called **eServer zSeries Application Assist Processor (zAAP)**. The zSeries Application Assist Processor is also known as an IFA (Integrated Facility for Applications). The term IFA often appears in panels, messages, and other z/OS information relating to the zSeries Application Assist Processor, including this publication.

RMF provides measurements for zAAP activity by enhancing the following tables:

Monitor III VSAM data set support
- ERBASIG3
- ERBENCG3
- ERBUWDG3

Monitor III data reporter tables
- ERBENCT3
- ERBPRCT3
- ERBSYST3

What's new in z/OS V1R5

64-bit API support
With the introduction of the z/Architecture®, applications can run in 64-bit addressing mode. The APIs that RMF offers to application programs to retrieve performance data for their own purposes can now be called by 31-bit callers and 64-bit callers using alternate entry points.

Monitor III VSAM Data Set Support
The following tables have been updated:
- ERBGEIG3
- ERBREDG3

What's new in z/OS V1R4

> **Note:**
> An appendix with z/OS product accessibility information has been added.

Monitor III VSAM Data Set Support
Table ERBCFIG3 describes the structure of records containing coupling facility information.

Table ERBCATG3 describes the structure of records containing cache data information.

Monitor III Data Reporter Tables
Table ERBCDCT3 for the Monitor III **CPC Capacity** report has been renamed to ERBCPCT3 and entries have been deleted from this table.

RMF Phase Driver Table ERBPHDS3 has been updated.

Monitor III VSAM Data Set Support
The following tables have been updated:

- ERBRCDG3
- ERBASIG3
- ERBCPUG3
- ERBDVTG3
- ERBGEIG3
- ERBUWDG3

LDAP Directory Structure
Object class `rmfCPC` has been added to the LDAP directory structure.

Chapter 1. SMF records

This chapter covers the following items:
- Summary of all RMF/SMF record types
- How to archive and print SMF records
- How to obtain SMF records directly

Overview

Each SMF record contains information similar to the contents of the corresponding formatted report. For each system activity that you select, RMF collects data and formats an SMF record to hold the data it collects.

Some totals, averages, and percentages are not explicitly contained in the SMF records, but are calculated from the SMF data. For elaboration of particular fields, see the descriptions of the corresponding fields in the printed report descriptions in *z/OS RMF Report Analysis*.

Also, each SMF record produced by RMF is described in *z/OS MVS System Management Facilities (SMF)*.

RMF does not generate reports from SMF records type 72, subtypes 2 or 4. However, these records are available for user-written reports.

Define the SMF record types and subtypes to be written in the SMFBUF option, which you can specify:
- In the PARM field of the RMF cataloged procedure
- On the system command START RMF
- On the system command MODIFY RMF

The record types and the corresponding RMF measurement activities are:
- Record type 70 has the following subtypes:
  - Subtype 1 – CPU and PR/SM™ activity
  - Subtype 2 – Cryptographic processor activity
- Record Type 71 – Paging activity
- Record type 72 has the following subtypes:
  - Subtype 3 – Workload activity
  - Subtype 4 – Storage data
- Record Type 73 – Channel path activity
- Record type 74 has the following subtypes:
  - Subtype 1 – Device activity
  - Subtype 2 – XCF activity
  - Subtype 3 – OMVS Kernel activity
  - Subtype 4 – Coupling facility activity
  - Subtype 5 – Cache subsystem activity
  - Subtype 6 – Hierarchical file systems statistics
  - Subtype 7 – FICON director statistics
  - Subtype 8 – Enterprise disk system statistics
- Record Type 75 – Page/Swap data set activity
- Record Type 76 – Trace activity
- Record Type 77 – Enqueue activity

- Record type 78 has the following subtypes:
  - Subtype 2 – Virtual storage activity
  - Subtype 3 – I/O queuing activity
- Record type 79 has the following subtypes for Monitor II snapshot data:
  - Subtype 1 – Address space state data
  - Subtype 2 – Address space resource data
  - Subtype 3 – Central storage/processor/SRM
  - Subtype 4 – Paging
  - Subtype 5 – Address space SRM data
  - Subtype 6 – Reserve data
  - Subtype 7 – Enqueue contention data
  - Subtype 9 – Device activity
  - Subtype 11 – Paging data set activity
  - Subtype 12 – Channel path activity
  - Subtype 14 – I/O queuing activity

You find details about which monitor is writing what SMF records in the *z/OS RMF User's Guide*.

SMF record format

Programming Interface information

Depending on the feedback options you select, RMF can write the SMF records to the SMF data set, use the data in the record to generate a printed report, or both. Regardless of the options you select, the format of the SMF record is the same.

Each SMF record that RMF generates consists of the following sections:

1. **SMF common header**, which identifies the record length, the record type, the time and date, the SMF system identifier, the subsystem identifier (always RMF), and the record subtype (if required). It also describes the other sections in the record. Each section is identified by its offset, the length of the section, and the number of such sections in the record. These offset/length/number triplet pointers define the structure of the rest of the record.
2. **RMF product section**, which includes information such as the RMF version number, the start time of the interval, the length of the interval, the length of the sampling cycle, and interval synchronization data. The RMF product section is the same in all records.
3. **Control section**, which contains general one-time data for RMF to use to produce any requested report. The contents of the section depend on the record type. Some records do not require a control section, while others require more than one.
4. **Data section**, which includes the specific data gathered during the interval. The format and the number of the data sections depend on the record type and the data collected. For example, there would be one data section for each device included in the type 74 record, I/O device activity.

With this format, the SMF records that RMF generates can change to incorporate any new or modified data without creating incompatibilities. The key factors in allowing for compatible change are the grouping of similar data in one section and the use of the offset/length/number triplet pointers to access the data stored in each section. Figure 1 shows the general format of the SMF records that RMF generates. The figure shows both the pointer structure and the storage layout for the sections.

Also, you can access fields in the SMF common header and the RMF product section by either a general name or a specific name. For example, you can access the interval start time in a type 70 record by either its general name (SMFIST) or its specific name (SMF70IST). Thus, code that processes all records can use the general name while code that processes only a specific record type can use the specific name.

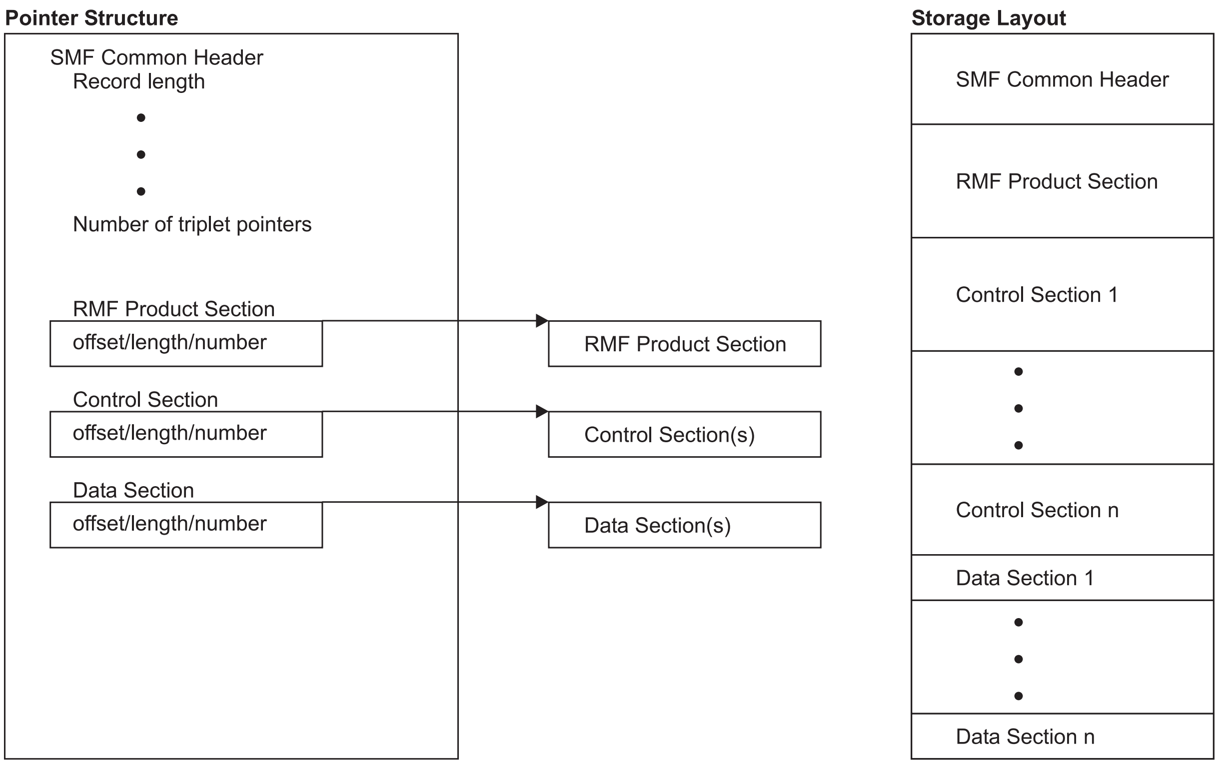


Figure 1. SMF Record Format

If your installation has existing data reduction programs that use SMF record input, check the SMF record formats carefully to determine what changes are required. Note that using the SMF record mapping macro instructions supplied by RMF is the most flexible way to access the contents of the SMF records your programs require. When you use the mapping macros, usually only a re-assembly of your program is required to incorporate changes to the record format.

The SMF record mapping macro instruction is ERBSMFR. Its format is:

```
ERBSMFR(nn1[,nn....1])
```

where nn identifies the type(s) of the SMF record(s) you want to map. Note that the parentheses are required only when two or more SMF record types are specified.

If you specify ERBSMF, the macro generates a mapping of the SMF common header and the RMF product section using only the general names.

The mapping macros reside in SYS1.MACLIB.

Because RMF can generate spanned SMF records – particularly when I/O device activity is measured – correct DCB parameters are important. Do not override the DCB parameters in the data set label by specifying DCB parameters on JCL statements. However, when using unlabeled tape the JCL describing an input SMF record data set should specify RECFM=VBS and a logical record length (LRECL) that is at least equal to the length of the longest record.

End of Programming Interface information

Archived performance data

You may find it useful to archive the performance data collected in the SMF records produced by RMF. You can use this data to study trends or to evaluate the impact of a system change. Because of system changes and/or RMF changes, the archived data recorded by various versions or releases of RMF is not always the same. The SMF record level change number field in all RMF SMF records lets you process any SMF record changes that may result from later RMF releases.

RMF version numbers

Programming Interface information

The Postprocessor reads the RMF version number of each SMF record in the input stream. This number appears in field name SMFxxMFV, where xx is the record number, the field contains one of the following values:

- X'610F' for an SMF record produced by OS/390 2.10.0 RMF

  This version number is also correct if RMF is running under z/OS V1R1.
- X'712F' for an SMF record produced by z/OS V1R2, V1R3 or V1R4 RMF
- X'715F' for an SMF record produced by z/OS V1R5 or V1R6 RMF
- X'717F' for an SMF record produced by z/OS V1R7 RMF
- X'718F' for an SMF record produced by z/OS V1R8 RMF

When the version number indicates that the record was produced by an earlier version or release of RMF, the Postprocessor converts the record to the current RMF format. A converted record, however, is not exactly the same as a current record. The major differences are:

- Fields for data that only the current version of RMF collects contain blanks or zeroes in the converted record.
- Fields for data that will not be collected anymore are omitted.
- The converted record contains a flag that indicates that it is a converted record, but RMF does preserve the original record version number.

These differences will also be reflected accordingly in the reports.

End of Programming Interface information

Printing SMF records

You might occasionally find it necessary to print the SMF records RMF produces. Printed records are useful, for example, when designing and implementing a user-written record processing program or when diagnosing problems with RMF reports. There are two ways to print the records:

- The standard utility program IDCAMS - it can print all SMF records in dump format.
- The RMF utility program ERBSCAN running under ISPF - it can format all SMF/RMF records in record-type-specific sections.

Using the IDCAMS utility

A sample of the JCL needed to print SMF records follows. The first step (SELECT) limits the amount of output to the record types or time frames that you need. If you want to print the entire data set, use only the second step (PRINT), defining the data set with the SMF records. These JCL statements and SMF dump parameters select and print SMF record type 74 that were written from 19:00 AM until 19:15 AM on April 3, 2011.

```
//SELECT    EXEC  PGM=IFASMFDP
//SYSPRINT  DD    SYSOUT=A
//IN        DD    DSN=data set containing SMF records
//OUT       DD    DSN=&&RMFREC,DISP=(NEW,PASS),UNIT=SYSDA
//SYSIN     DD    *
  INDD(IN,OPTIONS(DUMP))
  OUTDD(OUT,TYPE(74))
  START(1900)
  END(1915)
  DATE(20110403,20110403)
/*
//PRINT     EXEC  PGM=IDCAMS
//SYSPRINT  DD    SYSOUT=A
//RMFREC    DD    DSN=&&RMFREC,DISP=(OLD,PASS)
//SYSIN     DD    *
  PRINT   INFILE(RMFREC)
/*
```

z/OS MVS System Management Facilities (SMF) contains more information on the IFASMFDP dump program. *z/OS DFSMS Access Method Services for Catalogs* contains more information about IDCAMS.

Because you do not specify the format on the PRINT statement, the format defaults to DUMP. The records are printed in a dump format. Figure 2 is an example of the SMF record dump format. The offsets are in the left column, and the right side of the dump contains a printable section to help find the fields of interest. Note that the PRINT utility does not include the record length and segment descriptor fields in its output. As a result, a field shown at offset 4 in an SMF record in *z/OS MVS System Management Facilities (SMF)* appears at offset 0 in the formatted dump. You must adjust subsequent offsets accordingly to refer back and forth from the formatted dump to the printed SMF records in *z/OS MVS System Management Facilities (SMF)*.

```
IDCAMS  SYSTEM SERVICES                                      TIME: 13:28:15       04/28/11    PAGE      2

LISTING OF DATA SET -SYS1.SW.SMFIO

RECORD SEQUENCE NUMBER - 1
000000  1E02004E A56D0100 117FE5E2 D7D4                                          *...+._..."VSPM               *

RECORD SEQUENCE NUMBER - 2
000000  DF4A0069 A70B0100 094FD6E2 F0F4D9D4   C6400002 00050000 00000044 00680001   *.........|OS04RMF .............*
000020  000000AC 001C0001 000000C8 0038000E   000003D8 00580040 000019D8 002C00C0   *...........H.......Q... ...Q...{*
000040  606FD9D4 C6404040 40400185 900F0100   094F1500 000F0000 0000005A 00003000   *-?RMF     .......|.........!....*
000060  40404040 0009999F E5C5F0F2 F0F6F0F0   03E00438 B3D6ECD1 1A600000 00001AD2   *    ....VE020600.\...O.J.-.....K*
000080  74800000 00000000 00000000 03840DD4   B3D6ECD1 1A600000 E2D7D3C5 E7F0F140   *..............M.O.J.-..SPLEX01 *
0000A0  D6E2F0F4 40404040 00000000 00000000   00000000 00000000 00000000 00000000   *OS04    ........................*
0000C0  00000000 D6E2F0F1 40404040 00800000   00000008 00000000 00000000 00001B58   *....OS01    ....................*
0000E0  00000000 00000000 00000000 00000000   00000000 40404040 40404040 D6E2F0F1   *....................        OS01*
000100  40404040 00400000 00000002 00000000   00000000 00000ABE 00000000 00000000   *    . ..........................*
000120  0000007A 00000000 00003FBC C4C5C6C1   E4D3E340 D6E2F0F1 40404040 00400000   *...:........DEFAULT OS01    . ..*
000140  00000006 00000000 00000000 00001676   00000000 00006889 00000000 00000000   *................................*
```

Figure 2. Dump Format of SMF Record

Using the ERBSCAN utility

You can use the ERBSCAN utility to display RMF records directly under ISPF.

```
ERBSCAN data-set-name
```

```
VIEW       SYS11119.T130441.RA000.BTEU.R0500089           Columns 00001 00072
Command ===>                                                  Scroll ===> HALF
****** ***************************** Top of Data ******************************
000001 1z/OS V1R13 RMF ERBMFSCN Version 6 (28 May 1999) - SCAN SMF dataset
000002
000003  SMF dataset characteristics:
000004  RECFM    : VBS
000005  LRECL    : 32760
000006  BLKSIZE  : 27000
000007  DATASETS : 1
000008  DSNAME(S): SYS1.SW.SMFIO
000009  DATE/TIME: 2011 April 28     13:04:41.580
000010
000011
000012 1Rec-Num Type    RecLn SMFDate  SMFTime  RMFDate  RMFTime  Int-Len   SMF
000013  ------- ------- ----- ----------------- ----------------- --------- ---
000014        1 002        18 2011.117 14:19:01                             VSP
000015        2 074.002 15064 2011.094 19:14:00 2011.094 18:59:00 15:00:000 OS0
000016        3 074.003   412 2011.094 19:14:00 2011.094 18:59:00 15:00:000 OS0
000017        4 073.001 18680 2011.094 19:14:00 2011.094 18:59:00 15:00:090 OS0
000018        5 074.004  1912 2011.094 19:14:00 2011.094 18:59:00 15:00:000 OS0
000019        6 074.004  3816 2011.094 19:14:00 2011.094 18:59:00 15:00:000 OS0
000020        7 078.003 29976 2011.094 19:14:00 2011.094 18:59:00 15:00:090 OS0
000021        8 074.001 15520 2011.094 19:14:03 2011.094 18:59:00 15:00:090 OS0
000022        9 074.001 32604 2011.094 19:14:03 2011.094 18:59:00 15:00:090 OS0
```

Figure 3. ERBSCAN - Display RMF Record List

The function ERBSHOW is part of ERBSCAN to format a specific record. You call ERBSHOW with specifying a record number, for example:

```
COMMAND===> ERBSHOW 31
```

This leads to the display of the specified record showing the different sections as they are defined in the record:

```
VIEW       SYS11119.T132135.RA000.BTEU.R0500097            Columns 00001 00072
Command ===>                                                  Scroll ===> HALF
****** ***************************** Top of Data ******************************
000001 Record Number 31: SMF Record Type 74(1) - RMF Device Activity
000002 ==============================================================
000003
000004 -> SMF record header
000005 ====================
000006
000007    SMF record length      : 32604
000008    SMF segment descriptor : '0000'X
000009    SMF system indicator   : '11011111'B
000010    SMF record type        : 74
000011    SMF record time        : 19:14:03
000012    SMF record date        : 11.094
000013    SMF system id          : OS04
000014    SMF subsystem id       : RMF
000015    SMF record subtype     : 1
000016
000017 -> RMF header extension
000018 =======================
000019
000020    Number of triplets     : 3
000021
000022    Section  1 offset      : '00000034'X
000023    Section  1 length      : '0084'X
000024    Section  1 number      : 1
```

Figure 4. ERBSHOW - Display RMF Record Header

Scrolling forward leads to the different sections of the record.

```
VIEW       SYS11119.T132135.RA000.BTEU.R0500097            Columns 00001 00072
Command ===>                                                  Scroll ===> HALF
000058 -> Device Data Section (197)
000059 ============================
000060
000061    #1:   +0000:  2F8800F1 0001C4C5 F2C6F8F8 3030200F  * h 1  DE2F88    *
000062          +0010:  00000001 00000000 00000000 00000000  *                *
000063          +0020:  00000000 00000000 00000000 00000000  *                *
000064          +0030:  00000000 00000000 00000000 00000384  *               d*
000065          +0040:  00000000 00000000 00000000 00000000  *                *
000066          +0050:  00000000 00000000 40404040 40404040  *                *
000067          +0060:  00000000 F3F3F9F0 F3404040 F2F1F0F5  *    33903   2105*
000068          +0070:  40404040 00000000 20000000 984040F2  *             q  2*
000069          +0080:  F1F0F540 4040C9C2 D4F7F5F0 F0F0F0F0  *105   IBM7500000*
000070          +0090:  F0F0F1F4 F0F7F909 08000000 00000000  *0014079         *
000071          +00A0:  00000000                             *                *
000072
000073    #2:   +0000:  2F8900F1 0001C4C5 F2C6F8F9 3030200F  * i 1  DE2F89    *
000074          +0010:  00000001 00000000 00000000 00000000  *                *
000075          +0020:  00000000 00000000 00000000 00000000  *                *
000076          +0030:  00000000 00000000 00000000 00000384  *               d*
000077          +0040:  00000000 00000000 00000000 00000000  *                *
000078          +0050:  00000000 00000000 40404040 40404040  *                *
000079          +0060:  00000000 F3F3F9F0 F3404040 F2F1F0F5  *    33903   2105*
000080          +0070:  40404040 00000000 20000000 984040F2  *             q  2*
000081          +0080:  F1F0F540 4040C9C2 D4F7F5F0 F0F0F0F0  *105   IBM7500000*
000082          +0090:  F0F0F1F4 F0F7F909 09000000 00000000  *0014079         *
000083          +00A0:  00000000                             *                *
```

Figure 5. ERBSHOW - Display Device Data Section

Obtaining SMF record data directly

Programming Interface information

The RMF data interface service for Monitor II allows you to directly access SMF record data from storage in real time, rather than through SMF. Application programs can easily access SMF record data. The service provides easy access to SMF data for application programs. SMF record type 79, and the Monitor II header information for system CPU utilization and system demand paging rate, are supported.

To use the RMF data interface service, invoke the module ERBSMFI with the registers and parameters described in "Parameter list contents."

Note: Do not link the module ERBSMFI into your application program. Code the program to call ERBSMFI at run time. How to do this depends on the programming language you use:

- In Assembler, use LOAD or LINK macros
- In PL/I, use FETCH and RELEASE
- In C, use the fetch built-in function

The service returns only *one* record to the caller, which contains all the data. There is no 32K size limit; that is, the record is not broken up into 32K records.

The caller must be in 31-bit addressing mode and can run unauthorized.

Note that for some of the records, Monitor I must be running. These are as follows:
Subtype 9 - Device activity
Subtype 11 - Paging activity
Subtype 14 - I/O queuing activity

For more information about SMF record type 79, see "SMF record type 79" on page 140.

Registers at entry

The contents of the registers on entry to this service are:

Register	Contents
0	Not used
1	Parameter list address
2-12	Not used
13	Standard save area address
14	Return address
15	Entry point address of ERBSMFI

Parameter list contents

The parameter list passed by the caller to the RMF Monitor II data interface service contains nine fullword pointers, which contain the addresses of the following parameters:

Parameter 1
Fullword. Request type:

1 Parameter list contains 7 parameters

2 Parameter list contains 8 parameters

3 Parameter list contains 9 parameters

4 Parameter list contains 10 parameters

5 Parameter list contains 11 parameters

Parameter 2
Fullword. SMF record type requested, of which only type 79 is supported.

Parameter 3
Fullword. SMF record subtype requested.

Parameter 4
Buffer where the SMF record output is returned. Only one record is returned. See “Output” on page 10.

Parameter 5
Fullword. Length of the SMF record buffer.

To determine valid record lengths, see *z/OS MVS System Management Facilities (SMF)*. For address space related SMF record type 79 subtypes 1, 2, and 5, you must provide enough space for ASVTMAXU users. RMF does not return partial data. For other SMF record type 79 subtypes, RMF returns partial data if the buffer is not long enough.

Parameter 6
Fullword. Returns the system CPU utilization of standard CPs.

Parameter 7
Fullword. Returns the system demand paging rate.

Parameter 8
Input area which can hold the options used to generate the Monitor II reports.

The area starts with a 2-byte length field followed by the options. If the length field is initialized with 0, the default options are taken:

Subtype	Command
1	ASD(A,A,A)
2	ARD(A,A,A)
3	SRCS
4	SPAG
5	ASRM(A,A,A)
6	SENQR(ALLVSER)
7	SENQ(S)
9	DEV(NUMBER(0:FFFF))
11	PGSP(PAGE)
12	CHANNEL
14	IOQUEUE(DASD)

This parameter allows you to pass certain report options to the Monitor II data gatherer when parameter 1 contains the request type **2** or above. Please, refer to the *z/OS RMF User's Guide*, chapter "*Snapshot Reporting with Monitor II*", for other options. Use the display-session syntax described there.

Parameter 9
Fullword. Returns the MVS/SRM CPU utilization of standard CPs.

This parameter is returned for request type **≥3**.

Parameter 10
Fullword. Returns the system CPU utilization of zAAPs.

This parameter is returned for request type **≥4**.

Parameter 11 Fullword. Returns the system CPU utilization of zIIPs.

This parameter is returned for request type **≥5**.

Example:

To generate data for the Monitor II Device Activity report for all addresses in the range 0000 to 2FFF, you would have to issue the command:

```
DEV NUM(0000:2FFF)
```

You can specify this command with the following parameter list:

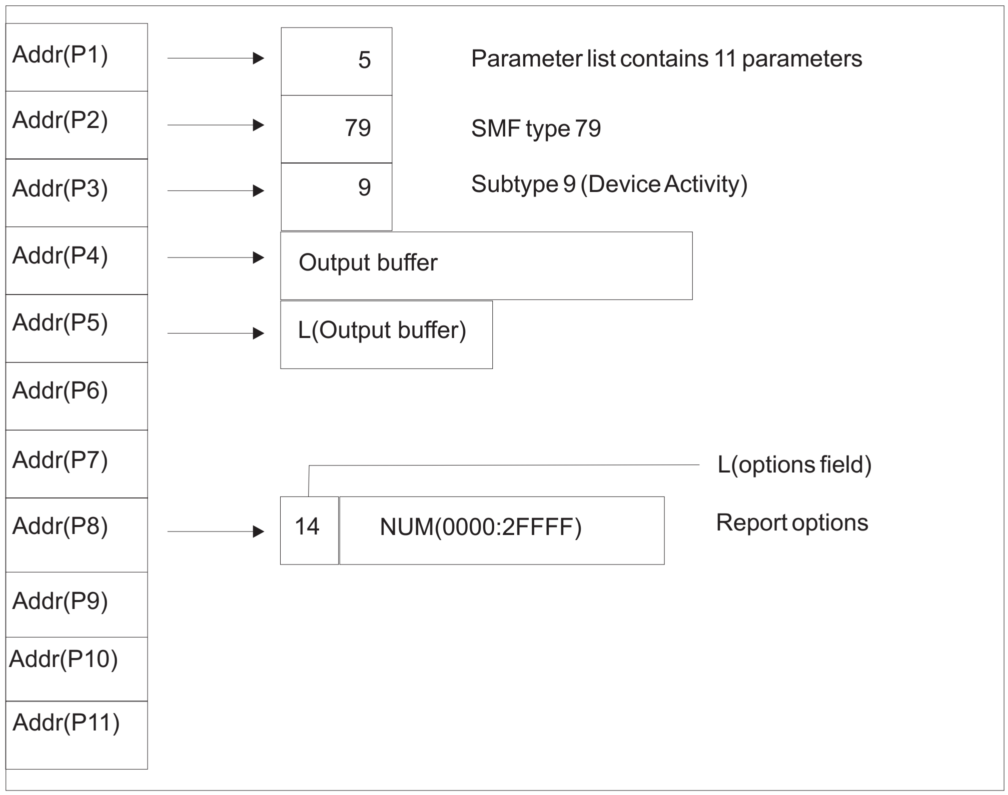


Figure 6. Parameter list for a command to obtain SMF record data

Output

The following are considerations for the output parameters:

Parameter 4 Contains the one SMF record that is returned with all of the data for the system. The SMFxxLEN field contains the length of the input buffer, not the actual length of the record. If the buffer is over 64K, the record contains X'FFFF'. If necessary, you can calculate the actual length of the record from the descriptor fields in the record. The date and time fields (SMF79DTE and SMF79TME fields, respectively) contain zeroes.

In case RMF was not started since the last IPL, the following fields are set to these values:

SMF79IML X'FF'

SMF79PTN X'FF'

SMF79FLG	LSB (bit 7) off
SMF79PRF	Bits 1 and 2 off

Parameter 6
Contains the current average standard CP utilization percent as a binary fullword in the area provided. If RMF cannot determine the CPU utilization percent on a PR/SM system because the Monitor I CPU report is not active, RMF returns a value of –1 (FFFFFFFF).

Parameter 7
Contains the page-ins per second rate as a binary fullword in the area provided. This rate is for demand paging to DASD only. It excludes swap-ins, VIO (virtual input/output), and hiperspaces.

Parameter 9
Contains the MVS view of the CPU utilization if Monitor I CPU gathering is active. Otherwise it is filled with the SRM view of the CPU utilization (source is CCVUTILP).

Parameter 10
Contains the current average zAAP utilization percent as a binary fullword in the area provided. If RMF cannot determine the zAAP utilization percent on a PR/SM system because the Monitor I CPU report is not active, RMF returns a value of –1 (FFFFFFFF).

Parameter 11
Same as Parameter 10, but for zIIPs.

Return codes

Upon return from this service, register 15 provides return codes listed in Table 3.

Table 3. Return Codes for the Monitor II Data Interface Service

Return Code (Decimal)	Description
0	Normal completion, data returned.
4	Incorrect syntax in parameter string.
8	Incorrect entry code (internal error in ERBSMFI).
16	Data is currently not available. It may be available at another time. Try again later.
20	Recovery environment could not be established.
24	Syntax error.
28	Data could not all fit in the buffer. Part of the data is returned. To get complete data, use a longer SMF buffer.
32	Data is not available; Monitor I gatherer is not active.
36	Data is reinitialized; Monitor I interval ended.
40	Data is not available. System resource manager's (SRM) store channel path status (STCPS) facility is not active.
44	Data is not available. System is in goal mode.
48	No transaction data available.
60	Invalid I/O measurement level.
100	Input record type or subtype is not valid.
104	No data is returned; SMF record buffer is too short.
108	Request type is not known.
112	ESTAE routine had control.
116	RMF is not enabled to run on this system.

Table 3. Return Codes for the Monitor II Data Interface Service (continued)

Return Code (Decimal)	Description
120	Service IFAEDREG or IFAEDDRG for registration or deregistration returned with a code greater than 4.
124	The user is not authorized to access Monitor II data.

Coded example

The following Assembler code example calls the Monitor II data interface service to obtain SMF record type 79 subtype 2 (address space resource data).

```
         ICTL     1,71,20
         PRINT    ON,GEN
EXSMFI   CSECT
         STM      R14,R12,12(R13)     Save entry regs
         LR       R12,R15             Set base from entry point
         USING    EXSMFI,R12          Tell asmblr of prcdr base
         LA       R2,SAVEAREA         Ptr to save area
         ST       R13,4(,R2)          Save old save in new area
         ST       R2,8(,R13)          Save new as forward of last
         LR       R13,R2              Point at new
* Get storage for SMF record buffer
         LA       R3,R792RLEN         Length of data section
         L        R4,CVTPTR           Address of CVT
         USING    CVT,R4
         L        R5,CVTASVT          ASVT address
         USING    ASVT,R5
         M        R2,ASVTMAXU         Multiply by maximum users
         DROP     R4                  CVT no longer needed
         DROP     R5                  ASVT no longer needed
         A        R3,HDRLEN           Add length of record headers
         SR       R4,R4               Subpool 0
         GETMAIN  RU,LV=(3),SP=(4)    Get storage
         ST       R1,BUFFER           Buffer address to parm list
         ST       R3,BUFLEN           Length to parm list
* Call ERBSMFI to create the record
         LA       R1,PARMLIST         Parameter to reg 1
         LINK     EP=ERBSMFI
*
* Check the return code and process the record here
*
         L        R2,BUFFER           Get ptr to buffer start
         L        R3,BUFLEN           Get buffer length
         SR       R4,R4               Subpool zero
         FREEMAIN RU,LV=(3),A=(2),SP=(4)
         L        R13,4(,R13)         Point at old save area
         SR       R15,R15             Set return code
         L        R14,12(,R13)        Restore return register
         LM       R0,R12,20(R13)      Restore all the rest
         BR       R14                 Return to caller
SAVEAREA DS       CL72                Save area
PARMLIST DC       A(REQTYPE)          Pointer to request type
         DC       A(RECTYPE)          Pointer to record type
         DC       A(SUBTYPE)          Pointer to subtype
BUFFER   DS       A                   Pointer to output buffer
         DC       A(BUFLEN)           Pointer to buffer length
         DC       A(CPUUTL)           Pointer to CPU utilization
         DC       A(DPR)              Pointer to demand paging rate
REQTYPE  DC       F'1'                Request type
RECTYPE  DC       F'79'               Record type 79
SUBTYPE  DC       F'2'                Subtype for ARD report record
BUFLEN   DS       F                   Length of SMF record buffer
CPUUTL   DS       F                   Return area for CPU util.
```

```
DPR      DS       F                   Return area for demand paging
HDRLEN   DC       A(HLEN+PLEN+CLEN)   Header length
*                                                                    *
**********************************************************************
*        Patch Area                                                  *
**********************************************************************
PATCH    DC       64S(*)
*
         LTORG
*
         PRINT    NOGEN
* SMF record 79 mapping
         ERBSMFR  79
* Record lengths
SMF79HDR DSECT
HLEN     EQU      *-SMF79HDR
SMF79PRO DSECT
PLEN     EQU      *-SMF79PRO
R79CHL   DSECT
CLEN     EQU      *-R79CHL
EXSMFI   CSECT
* System control block mappings
         CVT      DSECT=YES,LIST=NO
         IHAASVT  DSECT=YES,LIST=NO
* Registers
R0       EQU      0
R1       EQU      1
R2       EQU      2
R3       EQU      3
R4       EQU      4
R5       EQU      5
R6       EQU      6
R7       EQU      7
R8       EQU      8
R9       EQU      9
R10      EQU      10
R11      EQU      11
R12      EQU      12
R13      EQU      13
R14      EQU      14
R15      EQU      15
         END      EXSMFI
```

End of Programming Interface information

Chapter 2. RMF sysplex data services

The information in this chapter describes callable services (available for 31-bit and 64-bit environments) that enable you to access sysplex data:

- ERBDSQRY/ERBDSQ64 - RMF Query Available Sysplex SMF Data Service
- ERBDSREC/ERBDSR64 - RMF Request Sysplex SMF Record Data Service
- ERB2XDGS/ERB2XD64 - RMF Monitor II Sysplex Data Gathering Service
- ERB3XDRS/ERB3XD64 - RMF Monitor III Sysplex Data Retrieval Service

This chapter describes the CALL statements that invoke RMF sysplex data services. Each description includes a syntax diagram, parameter descriptions, and return code and reason code explanations with recommended actions. Return codes and reason codes are shown in decimal.

How to call sysplex data services

To use RMF sysplex data services, you issue CALLs that invoke the appropriate data service program. All of them are available as a set of APIs for the 31-bit environment, as well as for the 64-bit environment (see “How to call sysplex data services in 64-bit mode” on page 16). Each service program performs one or more functions and requires a set of parameters coded in a specific order on the CALL statement.

Do not link the data-services modules into your application program. Code the program to call the modules at run time. How you do this depends on the programming language you use:

- In Assembler, use LOAD or LINK macros
- In PL/I, use FETCH and RELEASE
- In C, use the fetch built-in function

The RMF supplied samplib contains three sample programs (written in C) that invoke these services (only in 31-bit mode):

- ERBDSMP1 calls ERBDSQRY/ERBDSREC
- ERBDSMP2 calls ERB2XDGS
- ERBDSMP3 calls ERB3XDRS

You might take one of the above sample programs as base for your own program MYERBSRV which you will code according to your requirements.

Example: The sample program ERBDSMP2 specifies ERBDSMX2 as exit. This program (an assembler program) is also part of the samplib and might be used as example on how to write an exit program.

Typically, one would not use sample exit ERBDSMX2 when running MYERBSRV, but would either use the RMF supplied exit ERB2XSMF which returns the complete SMF type 79 records, or one might write an own exit routine for a data reduction.

For ERB3XDRS (ERBDSMP3 sample program), the RMF supplied exit routine provided is ERB3XSOS (returns the complete Monitor III Set-of-Samples).

A sample JCL for compile, link and go could be set up as follows:

```
//    job record
//*
//PROCLIB  JCLLIB  ORDER=CBC.SCBCPRC
//*
//        EXEC EDCCLG,
//        INFILE='SYS1.SAMPLIB(ERBDSMP2)',
//        OUTFILE='loadlib(ERBDSMP2),DISP=SHR',
//        CPARM='OPTFILE(DD:OPTS)'
//COMPILE.OPTS DD *
SEARCH('SYS1.SAMPLIB')
SOURCE,NOLIST,OPTIMIZE,NOXREF,GONUMBER
```

Note:
For information about the RACF definitions needed to allow access to the sysplex data services, refer to *Controlling access to RMF data for the sysplex data services* in the *z/OS RMF User's Guide*.

How to call sysplex data services in 64-bit mode

With the introduction of the z/Architecture, applications can run in 64-bit addressing mode. The sysplex data services mentioned above can also be called by 64-bit callers using alternate entry points as described in Table 4:

Table 4. Sysplex data services

31-bit API	64-bit API
ERBDSQRY	ERBDSQ64
ERBDSREC	ERBDSR64
ERB2XDGS	ERB2XD64
ERB3XDRS	ERB3XD64

The parameters for the 64-bit API are identical to those for the 31-bit API. The 64-bit APIs may be called with parameters located above or below the 2 GB bar, except the answer area which must be located below the 2 GB bar.

Note:
Information provided for the 31-bit APIs is also valid for the 64-bit APIs, even though not explicitly mentioned. Exceptions from that rule are indicated where required.

ERBDSQRY - RMF Query available sysplex SMF data service

Programming Interface information

Call ERBDSQRY to request a directory of SMF record data available in the RMF Data Buffers on each system in the sysplex.

Write the CALL for ERBDSQRY as shown, coding all parameters in the specified order. Ensure that the values you assign to the parameters are in the format shown.

Table 5. ERBDSQRY Service

CALL ERBDSQRY	(answer_area_addr ,answer_area_alet ,answer_area_length ,request_type ,start_time ,end_time ,smf_record_type_info ,smf_record_type_list ,smf_system_name_info ,smf_system_name_list ,time_out ,return_code ,reason_code)

answer_area_addr
Specifies the address of the area where RMF returns the requested information. The area can be in the caller's primary address space or in an address or data space addressable through a public entry on the caller's Dispatchable Unit Access List (DU-AL).

Define *answer_area_addr* as pointer variable of length 4.

,answer_area_alet
Specifies the ALET of the answer area provided on the *answer_area_addr* parameter. If the area resides in the caller's primary address space, *answer_area_alet* must be 0.

Define *answer_area_alet* as unsigned integer variable of length 4.

,answer_area_length
Specifies the length of the answer area provided on the *answer_area_addr* parameter. If you do not provide enough length, RMF sets a return code and reason code, and places the length you need in the *answer_area_length* parameter.

Define *answer_area_length* as an unsigned integer variable of length 4.

,request_type
Specifies the ERBDSQRY request type. Specify one of the following values:

SMF Request information about SMF records of any type and subtype. Information will be returned about all SMF records whose time information, specified in the SMF record header, is within the time interval specified in the *start_time* and *end_time* parameters, that is:

```
C2S(start_time) ≤ (SMFxxDTE;SMFxxTME) ≤ C2S(end_time)
```

where C2S is the conversion function from character to SMF date and time format.

Note: This is the time the record was presented to SMF. For RMF-gathered data, it does not necessarily coincide exactly with the interval end time of the data collection interval.

The directory entries returned by ERBDSQRY contain the SMF record header plus a record token.

RMF Request information about SMF records of any RMF type and subtype. Information will be returned about all SMF records whose projected RMF measurement interval end time, specified in the

RMF product section, is within the time interval specified in the *start_time* and *end_time* parameters, that is:

```
C2T(start_time) ≤ (SMFxxGIE + SMFxxLGO) ≤ C2T(end_time)
```

where C2T is the conversion function from character to time-of-day (store clock) format.

Note: This is a theoretical value, it may not coincide with the actual RMF measurement interval (also part of the RMF product section of the SMF record).

The directory entries returned by ERBDSQRY contain SMF record header, RMF measurement interval information, plus a record token.

See “ERBDSQRY/ERBDSQ64 data section layout” on page 37.

Define *request_type* as character variable of length 3.

,start_time

Specifies the beginning of the time interval for which information is requested.

Define *start_time* as character variable of length 14 in the "sorted" format:

yyyy	mm	dd	hh	mm	ss

Figure 7. Format of the start time

If you want to omit this information, pass a value of 14 blanks. It will then default to the "oldest" SMF time found in any of the RMF Data Buffers at the time the service is called.

,end_time

Specifies the date and time of the end of the time interval information is requested for.

Define *end_time* as character variable of length 14 in the same "sorted" format as *start_time*.

If you want to omit this information, pass a value of 14 blanks. It will then default to the "newest" SMF time found in any of the RMF Data Buffers at the time the service is called.

,smf_record_type_info

Specifies the type of the list of SMF record types provided on the *smf_record_type_list* parameter. Specify one of the following values:

INCLUDE
The list of SMF record types provided on the *smf_record_type_list* parameter is an inclusion list. Information is requested for the listed SMF record types.

EXCLUDE
The list of SMF record types provided on the *smf_record_type_list* parameter is an exclusion list. Information is requested for all but the listed SMF record types.

ALL Information is requested for all SMF record types. The list of SMF record types provided on the *smf_record_type_list* parameter must start with an unsigned integer variable of length 4 set to a value of 0 (zero).

Define *smf_record_type_info* as a character variable of length 7. If you specify ALL, pad the string on the right with 4 blanks.

,smf_record_type_list

Specifies the list of SMF record types for which information is requested.

Define *smf_record_type_list* as an unsigned integer variable of length 4 (#rtypes) followed by an array of pairs of unsigned integers of length 2 (rt1... and st1...). The variable *#rtypes* specifies the number of array elements. Give *#rtypes* the value 0 (zero) to obtain information for all record types. The first number of each pair (rt1...) specifies the record type, and the second number of each pair (st1...) specifies the record subtype. For record types without subtypes, specify a subtype of 0.

Note: If you have specified **RMF** for *request_type*, record types outside the range 70 to 79 are ignored.

#rtypes	rt1	st1	rt2	st2	…	…

,smf_system_name_info

Specifies the type of the list of SMF system names provided on the *smf_system_name_list* parameter. Specify one of the following values:

INCLUDE

The list of SMF system names provided on the *smf_system_name_list* parameter is an inclusion list. Information is requested for systems with the listed SMF system names.

EXCLUDE

The list of SMF system names provided on the *smf_system_name_list* parameter is an exclusion list. Information is requested for all systems in the sysplex excluding the systems with the listed SMF system names.

ALL

Information is requested for all systems in the sysplex. The list of SMF system names provided on the *smf_system_name_list* parameter must start with an unsigned integer variable of length 4 set to a value of 0 (zero).

The list of SMF system names provided on the *smf_system_name_list* parameter is ignored. Information is requested for all systems in the sysplex.

Define *smf_system_name_info* as a character variable of length 7. If you specify ALL, pad the string on the right with 4 blanks.

,smf_system_name_list

Specifies the list of SMF system names information is requested for.

Define *smf_system_name_list* as an unsigned integer variable of length 4 that specifies the number of array elements, followed by an array of character variables of length 4.

#snames	#sn1	#sn2	…

,time_out
Specifies a time interval in seconds. If the time interval expires during the processing of the service, RMF returns to the caller with a corresponding return code and reason code and partial data.

Define *time_out* as a positive unsigned integer of length 4. Any other value will be overridden by a default value of 60.

,return_code
When ERBDSQRY completes, *return_code* contains the return code.

Define *return_code* as an unsigned integer variable of length 4.

For details see "Return codes and reason codes" on page 31.

,reason_code
When ERBDSQRY completes, *reason_code* contains the reason code.

Define *reason_code* as an unsigned integer variable of length 4.

For details see "Return codes and reason codes" on page 31.

End of Programming Interface information

ERBDSREC - RMF Request sysplex SMF record data service

Programming Interface information

Call ERBDSREC to request SMF record data from the RMF Data Buffers on each system in the sysplex. For each requested SMF record, include the record token, obtained from an earlier call of ERBDSQRY, on the list of record tokens passed as parameter to ERBDSREC.

Write the CALL for ERBDSREC as shown, coding all parameters in the specified order. Ensure that the values you assign to the parameters are in the format shown.

Table 6. ERBDSREC Service

CALL ERBDSREC	(answer_area_addr ,answer_area_alet ,answer_area_length ,rmf_record_token_list ,time_out ,return_code ,reason_code)

answer_area_addr
Specifies the address of the area to which RMF returns the requested information. The area can be in the caller's primary address space or in an address or data space addressable through a public entry on the caller's Dispatchable Unit Access List (DU-AL).

Define *answer_area_addr* as a pointer variable of length 4.

,answer_area_alet

Specifies the ALET of the answer area provided on the *answer_area_addr* parameter. If the area resides in the caller's primary address space, *answer_area_alet* must be 0.

Define *answer_area_alet* as an unsigned integer variable of length 4.

,answer_area_length

Specifies the length of the answer area provided on the *answer_area_addr* parameter. If you do not provide enough length, RMF sets a return code and reason code, and places the length you need in the *answer_area_length* parameter.

Define *answer_area_length* as unsigned integer variable of length 4.

,rmf_record_token_list

Specifies the list of record tokens for the requested SMF records.

Define *rmf_record_token_list* as an unsigned integer variable of length 4 that specifies the number of array elements, followed by an array of character of length 8.

#tokens	token1	token2	…

,time_out

Specifies a time interval in seconds. If the time interval expires during the processing of the service, RMF returns to the caller with a corresponding return code and reason code and partial data.

Define *time_out* as a positive unsigned integer of length 4. Any other value will be overridden by a default value of 60.

,return_code

When ERBDSREC completes, *return_code* contains the return code.

Define *return_code* as an unsigned integer variable of length 4.

For details see “Return codes and reason codes” on page 31.

,reason_code

When ERBDSREC completes, *reason_code* contains the reason code.

Define *reason_code* as an unsigned integer variable of length 4.

For details see “Return codes and reason codes” on page 31.

End of Programming Interface information

ERB2XDGS - RMF Monitor II sysplex data gathering service

Programming Interface information

Call ERB2XDGS to request Monitor II data according to the specified SMF record type 79 (Monitor II) subtype.

Write the CALL for ERB2XDGS as shown, coding all parameters in the specified order. For parameters that ERB2XDGS uses to obtain input values, assign values that are acceptable to ERB2XDGS.

Table 7. ERB2XDGS Service

CALL ERB2XDGS	(answer_area_addr ,answer_area_alet ,answer_area_length ,system_name ,data_gatherer_parm ,data_gatherer_parm_length ,exit_name ,exit_parm ,exit_parm_length ,time_out ,return_code ,reason_code)

answer_area_addr
Specifies the address of the area where RMF returns the requested information. The area can be in the calling program's primary address space, or in an address or data space addressable through a public entry on the calling program's dispatchable unit access list (DU-AL).

Define *answer_area_addr* as pointer variable of length 4.

,answer_area_alet
Specifies the ALET of the answer area provided on the *answer_area_addr* parameter. If the area resides in the calling program's primary address space, *answer_area_alet* must be 0.

Define *answer_area_alet* as unsigned integer variable of length 4.

,answer_area_length
Specifies the length of the answer area provided on the *answer_area_addr* parameter. If you do not provide enough space, RMF lets you know how much space you should have provided. The *answer_area_length* input/output parameter contains the length needed for the complete data.

Define *answer_area_length* as unsigned integer variable of length 4.

,system_name
Specifies the name of the system for which you are requesting information. This is the four character SMF system identification (SID). ***ALL** specifies that the request is to be sent to **all** systems in the sysplex.

Define *system_name* as character variable of length 4.

,data_gatherer_parm
Specifies the parameters for the Monitor II data gatherer on each system.

Define *data_gatherer_parm* as structure variable of variable length. The layout of the parameter area is as follows:

rty	sty	dg_options ...

where:

rty Specifies the SMF record type of the requested Monitor II data.

Define *rty* as unsigned integer variable of length 2.

sty Specifies the SMF record subtype of the requested Monitor II data.

Define *sty* as unsigned integer variable of length 2.

dg_options
Specifies options for the Monitor II data gatherer for the specified SMF record type and subtype.

Define *dg_options* as character variable of variable length, maximum 32.

You find a list of all subtypes in "Overview" on page 1.

Example

You want to receive data that is equivalent to the Monitor II command

```
SENQ D
```

This requires the following values for this parameter:

rty	SMF record type - **79**
sty	SMF record subtype for the `SENQ` - **07**
dg_options	Command option - **D**

This results in the value '7907D' for the data gatherer parameter.

,data_gatherer_parm_length
Specifies the length of the parameter string *data_gatherer_parm*.

Define *data_gatherer_parm_length* as unsigned integer variable of length 4.

,exit_name
Specifies the name of a data reduction exit routine that is invoked by RMF on each system from which data is requested. After the Monitor II data has been retrieved by RMF, this exit may call selected areas from the data to the answer area provided by RMF. These data areas are then combined into the answer area provided by the caller on the requesting system.

The data reduction exit routine ERB2XSMF, provided by IBM, copies the complete data gathered by the Monitor II data gatherer (SMF record type 79) to the answer area. ERB2XSMF has no exit parameters.

Define *exit_name* as character variable of length 8.

,exit_parm
Specifies a parameter string that may be passed to the routine specified in *exit_name*. Use this parameter to control the selection of Monitor II data areas to be returned to the caller.

Define *exit_parm* as character variable of variable length, maximum 32768.

,exit_parm_length
Specifies the length of the parameter string *exit_parm* that is passed to the routine specified in *exit_name*.

Define *exit_parm_length* as unsigned integer variable of length 4.

,time_out
Specifies a time interval in seconds. If this time interval expires during the processing of the service, RMF returns to the caller with a corresponding return and reason code and partial data.

Define *time_out* as unsigned integer variable of length 4.

The specification of a non-positive value will cause RMF to use a default value of 60.

,return_code
When ERB2XDGS completes, *return_code* contains the return code.

Define *return_code* as unsigned integer variable of length 4.

For details see "Return codes and reason codes" on page 31.

,reason_code
When ERB2XDGS completes, *reason_code* contains the reason code.

Define *reason_code* as unsigned integer variable of length 4.

For details see "Return codes and reason codes" on page 31.

End of Programming Interface information

ERB2XDGS data reduction exit routines

Programming Interface information

The exit routine specified in the **exit_name** parameter of the ERB2XDGS service is invoked on each system to which the ERB2XDGS request was directed. The routine is assumed to have the following attributes:

Location: JPA

State: Problem

Key: Any

Amode: 31

Rmode: Any

Dispatchable unit mode: Task

Address space control mode: AR

Cross Memory Mode: PASN=SASN=HASN

Serialization: Enabled, unlocked

Type: Reentrant, Refreshable

The exit is called by RMF as shown, with the parameters in the specified order.

Table 8. ERB2XDGS Exit Routine

CALL exit_name	(answer_area_addr ,answer_area_alet ,answer_area_length ,output_area_length ,input_data_address ,exit_parm ,exit_parm_length)

answer_area_addr
Specifies the address of the area where the exit routine may return the selected information. The area resides in a data space owned by the RMF address space.

Answer_area_addr is defined as pointer variable of length 4.

,answer_area_alet
Specifies the ALET of the answer area provided on the *answer_area_addr* parameter.

Answer_area_alet is defined as unsigned integer variable of length 4.

,answer_area_length
Specifies the length of the answer area provided on the *answer_area_addr* parameter. RMF provides an answer area in the length of the answer area the caller provided to ERB2XDGS, rounded to the next multiple of 4096. However, the data returned by the data reduction exit routine must fit into the answer area the caller provided to ERB2XDGS, including the common header and data headers created by RMF.

Answer_area_length is defined as unsigned integer variable of length 4.

,output_area_length
Specifies the length of the data that the exit routine provided. If this value is larger than *answer_area_length*, a return and reason code are set, indicating that the length of the answer area was not sufficient.

Output_area_length is defined as unsigned integer variable of length 4 and **must be set by the exit routine**.

,input_area_address
Specifies the address of the SMF record type 79 image in storage.

Input_area_address is defined as pointer variable of length 4.

,exit_parm
Specifies the parameter that has been provided for the exit routine by the caller of ERB2XDGS.

Exit_parm is defined as character variable of variable length.

,exit_parm_length
Specifies the length of the parameter string *exit_parm* that was passed to the exit routine.

Exit_parm_length is defined as unsigned integer variable of length 4.

End of Programming Interface information

ERB3XDRS - RMF Monitor III sysplex data retrieval service

Programming Interface information

Call ERB3XDRS to request a set-of-samples of Monitor III data from to the specified date and time range.

Write the CALL for ERB3XDRS as shown, coding all parameters in the specified order. For parameters that ERB3XDRS uses to obtain input values, assign values that are acceptable to ERB3XDRS.

Table 9. ERB3XDRS Service

CALL ERB3XDRS	(answer_area_addr ,answer_area_alet ,answer_area_length ,system_name ,data_retrieval_parm ,data_retrieval_parm_length ,exit_name ,exit_parm ,exit_parm_length ,time_out ,return_code ,reason_code)

answer_area_addr
: Specifies the address of the area to which RMF returns the requested information. The area can be in the calling program's primary address space or in an address or data space addressable through a public entry on the calling program's dispatchable unit access list (DU-AL).

  Define *answer_area_addr* as pointer variable of length 4.

,answer_area_alet
: Specifies the ALET of the answer area provided on the *answer_area_addr* parameter. If the area resides in the calling program's primary address space, *answer_area_alet* must be 0.

  Define *answer_area_alet* as unsigned integer variable of length 4.

,answer_area_length
: Specifies the length of the answer area provided on the *answer_area_addr* parameter. If you do not provide enough space, RMF lets you know how much space you should have provided. The *answer_area_length* input/output parameter contains the length needed for the complete data.

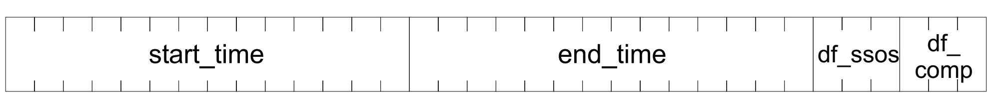
  Define *answer_area_length* as unsigned integer variable of length 4.

,system_name
: Specifies the name of the system for which information is being requested. This is the four-character SMF system ID (SID). ***ALL** specifies that the request is to be sent to **all** systems in the sysplex. However, only the systems with a running Monitor III data gatherer session are able to return the requested data.

  Define *system_name* as character variable of length 4.

,data_retrieval_parm
: Specifies the parameters for the retrieval of Monitor III data on each system.

  Define *data_retrieval_parm* as structure variable with a length of 34 bytes. This structure contains the start and end of the range for which data is requested, and parameters that define the format of the returned data. The layout of the 34-byte parameter area is as follows:



start_time
Specifies the date and time of the beginning of the time range for which information is requested.

Define *start_time* as a character variable of length 14 in “sorted” format.

yyyy	mm	dd	hh	mm	ss

If you want to omit this information, pass a value of 14 blanks. ERB3XDRS will then return information for one Monitor III MINTIME, ending with or containing the date and time specified in *end_time*. If this parameter is omitted as well, information for the latest available MINTIME is returned.

end_time
Specifies the date and time of the end of the time range for which information is requested.

Define *end_time* as character variable of length 14 in the same "sorted" format as *start_time*.

If you want to omit this information, pass a value of 14 blanks. ERB3XDRS will then return information for one Monitor III MINTIME, starting with or containing the date and time specified in *start_time*. If this parameter is omitted as well, information for the latest available MINTIME is returned.

df_ssos
Data format Single Set-Of-Samples - specifies whether or not the set-of-samples data should be returned as a combined set-of-samples (as opposed to a sequence of individual sets-of-samples).

YES the data is returned in a combined form, that is, the individual sets-of-samples are combined into one common set-of-samples.

NO the data is returned in individual sets-of-samples.

Define *df_ssos* as character variable of length 3. If you specify NO, pad the string on the right with a blank.

df_comp
Data format Compressed Set-Of-Samples - specifies whether or not the set-of-samples data should be returned in compressed format

YES the data is returned compressed (as it resides in the Monitor III data sets). This means that it will have to be decompressed using the RMF service ERB3RDEC.

NO the data is returned uncompressed

Define *df_comp* as character variable of length 3. If you specify NO, pad the string on the right with a blank.

,data_retrieval_parm_length
Specifies the length of the parameter string *data_retrieval_parm*.

Define *data_retrieval_parm_length* as unsigned integer variable of length 4.

,exit_name
Specifies the name of a data reduction exit routine that is invoked by RMF on each system from which data is requested. After the set-of-samples data has been retrieved by RMF, this exit may call selected areas from the set-of-samples to the answer area provided by RMF. These data areas are then combined into the answer area provided by the caller on the requesting system.

The data reduction exit routine ERB3XSOS, provided by IBM, copies the complete data retrieved from the Monitor III data gatherer (the set-of-samples data) to the answer area. ERB3XSOS has no exit parameters.

Define *exit_name* as a character variable of length 8.

,exit_parm
Specifies a parameter string that may be passed to the routine specified in *exit_name*. Use this parameter to control the selection of set-of-samples data areas that are to be returned to the caller.

Define *exit_parm* as a character variable of variable length, with a maximum of 32768.

,exit_parm_length
Specifies the length of the parameter string *exit_parm* that is passed to the routine specified in *exit_name*.

Define *exit_parm_length* as an unsigned integer variable of length 4.

,time_out
Specifies a time interval in seconds. If this time interval expires during the processing of the service, RMF returns to the caller with a corresponding return and reason code and partial data.

Define *time_out* as an unsigned integer variable of length 4.

The specification of a non-positive value will cause RMF to use a default value of 60.

,return_code
When ERB3XDRS completes, *return_code* contains the return code.

Define *return_code* as an unsigned integer variable of length 4.

For details see “Return codes and reason codes” on page 31.

,reason_code
When ERB3XDRS completes, *reason_code* contains the reason code.

Define *reason_code* as an unsigned integer variable of length 4.

For details see “Return codes and reason codes” on page 31.

End of Programming Interface information

ERB3XDRS data reduction exit routines

Programming Interface information

The exit routine specified in the **exit_name** parameter of the ERB3XDRS service is invoked on each system the ERB3XDRS request was directed to. The routine is assumed to have the following attributes:

Location:	JPA
State:	Problem
Key:	Any
Amode:	31
Rmode:	Any
Dispatchable unit mode:	Task
Address space control mode:	AR
Cross Memory Mode:	PASN=SASN=HASN
Serialization:	Enabled, unlocked
Type:	Reentrant, Refreshable

The exit is called by RMF as shown, with the parameters in the specified order.

Table 10. ERB3XDRS Exit Routine

CALL exit_name	(answer_area_addr ,answer_area_alet ,answer_area_length ,output_area_length ,input_data_address ,exit_parm ,exit_parm_length)

answer_area_addr
Specifies the address of the area to which the exit routine may return the selected information. The area resides in a data space owned by the RMF address space.

Answer_area_addr is defined as a pointer variable of length 4.

,answer_area_alet
Specifies the ALET of the answer area provided on the *answer_area_addr* parameter.

Answer_area_alet is defined as an unsigned integer variable of length 4.

,answer_area_length
Specifies the length of the answer area provided on the *answer_area_addr* parameter. RMF provides an answer area the same length as the answer area that the caller provided for ERB3XDRS, rounded to the next multiple of 4096. However, the data returned by the data reduction exit routine must fit into the answer area the caller provided for ERB3XDRS, including the common header and data headers created by RMF.

Answer_area_length is defined as an unsigned integer variable of length 4.

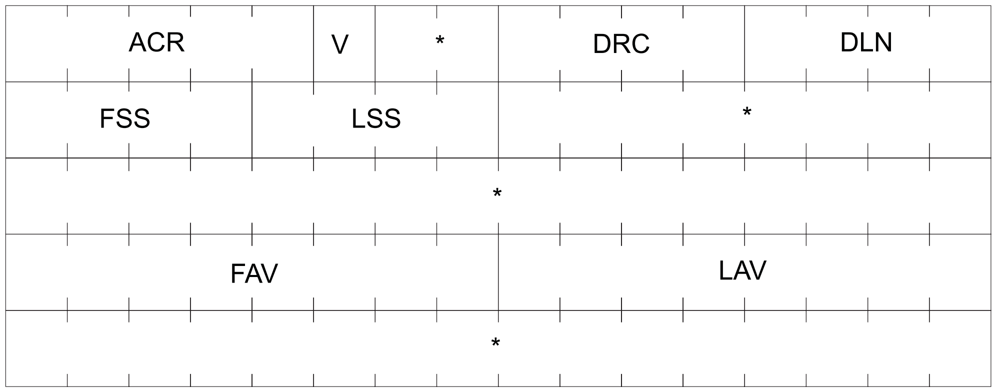
,output_area_length
Specifies the length of the data that is provided by the exit routine. If this value is larger than *answer_area_length*, a return and reason code is set, indicating that the length of the answer area is not sufficient.

Output_area_length is defined as an unsigned integer variable of length 4 and **must be set by the exit routine**.

,input_area_address
Specifies the address of the data reduction exit input data area. This data

area contains the Monitor III control block XMHG3 at offset 0, followed by zero or more sets-of-samples, each of them starting with the Monitor III control block SSHG3.

Input_area_address is defined as a pointer variable of length 4. Control block XMHG3 has the following format:



ACR (offset +00, length 5) Acronym of XMHG3, EBCDIC "XMHG3"

V (offset +05, length 1) Version of XMHG3

DRC (offset +08, length 4) Data return code. The possible codes are:

- **0** Successful data retrieval
- **4** Time out of range
- **8** Area too small
- **12** No data available
- **16** Severe error

DLN (offset +12, length 4) Total data length including XMHG3 itself

FSS (offset +16, length 4) Offset from XMHG3 to first set-of-samples header SSHG3

LSS (offset +20, length 4) Offset from XMHG3 to last set-of-samples header SSHG3

FAV (offset +40, length 8) Time in STCK format of first available data

LAV (offset +48, length 8) Time in STCK format of last available data

,exit_parm
Specifies the parameter for the exit routine that has been provided by the caller of ERB3XDRS.

Exit_parm is defined as character variable of variable length.

,exit_parm_length
Specifies the length of the parameter string *exit_parm* that is passed to the exit routine.

Exit_parm_length is defined as unsigned integer variable of length 4.

End of Programming Interface information

Return codes and reason codes

Programming Interface information

When the RMF Sysplex Data Services return control to your program, *return_code* contains the return code and *reason_code* contains the reason code.

Not every combination of return and reason codes applies to each of the services. The possible combinations are shown in Table 11.

The return and reason codes are grouped into classes indicating the severity of the situation that has been recognized. The classes are:

Successful (RC=0)
The operation was successful. The requested data has been stored in the answer area provided by the calling program

Information (RC=4)
The requested data may be inconsistent (ERB3XDRS and ERB3XD64 only)

Warning (RC=8)
The requested data could not be retrieved completely

Error (RC=12)
No data was returned, for example, because no RMF address space was active

Severe Error (RC=16)
The calling program invoked the service with invalid parameters or in an invalid mode

Unrecoverable Error (RC=20)
A problem has been detected within RMF processing. This code is normally accompanied by console messages, or a dump, or both. Refer to the explanations of the issued messages.

The following table identifies return code and reason code combinations, and recommends the action that you should take. Codes are decimal numbers. Applicable service routines are:

Q ERBDSQRY and ERBDSQ64
R ERBDSREC and ERBDSR64
2 ERB2XDGS and ERB2XD64
3 ERB3XDRS and ERB3XD64

Table 11. RMF Sysplex Data Services Return and Reason Codes (SMF Services)

Return Code	Reason Code	Service Routine	Meaning / Action
0	0	Q,R,2,3	**Meaning:** The operation was successful. The answer area contains the requested data.
			Action: Continue normal program execution.
8	8	-,-,-,3	**Meaning:** Warning - data could not be retrieved. For the specified date and time range, either partial data or no data at all could be retrieved by the ERB3XDRS service because time gaps have been detected in the gathered data.
			Action: Check the time range (*start_time* or *end_time*) parameters on the ERB3XDRS service and rerun the program.

Table 11. RMF Sysplex Data Services Return and Reason Codes (SMF Services) (continued)

Return Code	Reason Code	Service Routine	Meaning / Action
8	9	-,-,-,3	**Meaning:** Warning - VSAM retrieval errors occurred. For the specified date and time range, either partial data or no data at all could be retrieved.
			Action: Check the time range (*start_time* or *end_time*) parameters on the ERB3XDRS service and rerun the program.
8	13	-,-,-,3	**Meaning:** Warning - inconsistent data returned by ERB3XDRS. The WLM service policy has changed, or the IPS values have been modified.
8	14	-,-,-,3	**Meaning:** Warning - inconsistent data returned by ERB3XDRS. The RMF cycle time has changed.
8	15	-,-,-,3	**Meaning:** Warning - inconsistent data returned by ERB3XDRS. IPL detected.
8	30	Q,R,-,-	**Meaning:** Warning - timeouts detected. Due to timeout situations, ERBDSQRY or ERBDSREC could not return all the requested information.
			Action: Request a smaller a mount of information on one call of the RMF service.
8	31	-,R,-,-	**Meaning:** Warning - no such record. One or more requested SMF records were not available for ERBDSREC, either the SMF record data was overwritten by the wrap-around management of the data buffer or it never existed.
			Action: Ensure that the elapsed time between calls to ERBDSQRY and ERBDSREC is not too large, and that a valid token list is passed to ERBDSREC.
8	35	-,-,2,-	**Meaning:** Warning - defaults taken. Due to incorrectly specified Monitor II data gatherer options on the *dg_options* parameter of the ERB2XDGS service, the data gatherer decided to use the default options.
			Action: Correct Monitor II data gatherer options and rerun the program.
8	70	Q,R,-,-	**Meaning:** Warning - answer area too small. The answer area provided by the calling program was too small for the service to return all the requested information. The variable *answer_area_length* contains the length of the answer area you should have provided for this ERBDSQRY or ERBDSREC request.
			Action: Provide an answer area large enough to contain all the requested information.
12	0	Q,R,2,3	**Meaning:** Error - RMF Sysplex Data Server is not active.
			Action: Start the local RMF address space.
12	1	Q,R,2,3	**Meaning:** Error - System(s) inactive. None of the system(s) specified for the ERBDSQRY, ERB2XDGS, or ERB3XDRS services were active in the sysplex. For ERBDSREC, none of the record tokens specified belong to SMF records collected on systems that are currently active in the sysplex.
			Action: Check the system name list (*smf_system_name_list*, for ERBDSQRY), record token list (*rmf_record_token_list*, for ERBDSREC), or the system name (*system_name*, for ERB2XDGS and ERB3XDRS) parameter and rerun the program.
12	5	-,-,2,-	**Meaning:** Error - Monitor I interval ended. The Monitor I interval ended during the Monitor II data gathering phase while processing the ERB2XDGS request.
			Action: Rerun the program.
12	6	-,-,2,-	**Meaning:** Error - No RMF data available. No data is currently available that matches the specification in the *data_gathering_parm* parameter of the ERB2XDGS service.
			Action: Check the parameters of ERB2XDGS and rerun the program.
12	7	-,-,2,-	**Meaning:** Error - No Monitor I data gatherer. The Monitor I data gatherer was not active. However, for the data gathering of certain SMF record subtypes (record type 79, subtypes 8, 9, 11, 13, and 14) specified for the ERB2XDGS service, an active Monitor I session is required.
			Action: Verify Monitor I is active on the systems from which data is requested, and rerun the program.
12	8	-,-,-,3	**Meaning:** Error - data could not be retrieved. For the specified date and time range, no data could be retrieved by the ERB3XDRS service.
			Action: Check the time range (*start_time* or *end_time*) parameters on the ERB3XDRS service and rerun the program.

Table 11. RMF Sysplex Data Services Return and Reason Codes (SMF Services) (continued)

Return Code	Reason Code	Service Routine	Meaning / Action
12	9	-,-,-,3	**Meaning:** Error - VSAM retrieval errors occurred. For the specified date and time range, no data could be retrieved by the ERB3XDRS service.
			Action: Check the time range (*start_time* or *end_time*) parameters on the ERB3XDRS service and rerun the program.
12	16	-,-,-,3	**Meaning:** Error - no data returned by ERB3XDRS. No data available.
12	17	-,-,-,3	**Meaning:** Error - The Monitor III session is not active on the system specified on the *system_name* parameter of the ERB3XDRS service. If data was requested from all systems in the sysplex, no Monitor III session was found active in the sysplex.
			Action: Start Monitor III on the system(s) for which Monitor II data was requested. Check the system name parameter passed to the ERB3XDRS service.
12	18	-,-,-,3	**Meaning:** Error - no data returned by ERB3XDRS. Preallocated data sets unusable (detected at start of retrieval).
12	19	-,-,-,3	**Meaning:** Error - no data returned by ERB3XDRS. Preallocated data sets unusable (detected during data retrieval).
12	20	-,-,-,3	**Meaning:** Error - no data returned by ERB3XDRS. Too many reporters tried to get data from the in-storage buffer.
12	21	-,-,-,3	**Meaning:** Error - no data returned by ERB3XDRS. Retrieval from in-storage buffer failed.
12	22	-,-,-,3	**Meaning:** Error - no data returned by ERB3XDRS. No data in the in-storage buffer.
12	23	-,-,-,3	**Meaning:** Error - no data returned by ERB3XDRS. Not enough storage available to copy the requested data from the in-storage buffer.
12	25	-,-,2,-	**Meaning:** Error - SRM STCPS facility not available. The system resource manager (SRM) Store Channel Path Status (STCPS) facility is not available.
12	30	-,-,2,3	**Meaning:** Error - Timeout. Due to a timeout situation, ERB2XDGS or ERB3XDRS could not return the requested information.
			Action: Request a smaller amount of information on one call of the ERB2XDGS or ERB3XDRS service.
12	36	Q,-,-,-	**Meaning:** Error - no data returned by ERBDSQRY. No SMF data was found in the sysplex matching the specification provided by the *smf_start_time*, *smf_end_time*, *smf_record_type_info*, *smf_record_type_list*, *smf_system_name_info*, and *smf_system_name_list* parameters of the ERBDSQRY service.
			Action: Check the parameter specifications.
12	37	Q,R,-,-	**Meaning:** Error - All RMF Data Buffers for SMF data are inactive on the systems specified on the *smf_system_name_info* and *smf_system_name_list* parameters of the ERBDSQRY service. For ERBDSREC, an attempt was made to request SMF records from a system on which the RMF data buffer is inactive.
			Action: Start RMF Data Buffer on one or more systems in the sysplex. Check the list of system names passed to the ERBDSQRY service.
12	70	-,-,2,3	**Meaning:** Error - answer area too small. The answer area provided by the calling program was too small for the service to return all the requested information. The variable *answer_area_length* area contains the length of the answer you should have provided for this ERB2XDGS or ERB3XDRS request.
			Action: Provide an answer area large enough to contain all the requested information.
16	0	-,-,-,-	**Meaning:** Reserved for RMF internal use.
			Action: Not applicable.
16	41	Q,-,-,-	**Meaning:** Severe error - The calling program specified an invalid value for the request type (*request_type*). parameter for ERBDSQRY.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.

Table 11. RMF Sysplex Data Services Return and Reason Codes (SMF Services) (continued)

Return Code	Reason Code	Service Routine	Meaning / Action
16	42	Q,-,-,3	**Meaning:** Severe error - The calling program specified an invalid value for the interval/range start or end time (*start_time* or *end_time*) or parameter (YYYYMMDDHHMMSS) on the ERBDSQRY ERB3XDRS service. This includes wrong-formatted parameters and out-of-range or invalid dates.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	43	Q,-,-,-	**Meaning:** Severe error - The calling program specified an invalid value for the SMF record type (*smf_record_type_info*) parameter (INCLUDE/EXCLUDE/ALL) of the ERBDSQRY service.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	44	Q,-,-,-	**Meaning:** Severe error - The calling program specified an invalid value for the SMF system name (*smf_system_name_info*) parameter (INCLUDE/EXCLUDE/ALL) of the ERBDSQRY service.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	45	-,-,-,3	**Meaning:** Severe error - The calling program specified an invalid value for the data format (*df_ssos* or *df_comp*) subparameters (YES/NO) of the ERB3XDRS service.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	46	-,-,2,-	**Meaning:** Severe error - A bad SMF record type or subtype (*rty* or *sty*) was specified for the ERB2XDGS service.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	52	-,-,-,3	**Meaning:** Severe error - The calling program specified range start and end times with a difference greater than 9999 seconds in the (*start_time* and *end_time*) parameters of the ERB3XDRS service.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	53	Q,-,-,-	**Meaning:** Severe error - An invalid SMF record type or subtype was specified in the record type list (*smf_record_type_list*) for the ERBDSQRY service. Either the length of the list was negative, or a record type was out of the range of 0 to 255.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	54	Q,-,-,-	**Meaning:** Severe error - An invalid SMF system name was specified in the system name list (*smf_system_name_list*) for the ERBDSQRY service, or the length of the list was negative.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	55	Q,-,-,3	**Meaning:** Severe error - An invalid data time interval (*start_time* or *end_time*) was specified for the ERBDSQRY or ERB3XDRS service, i.e. the start time is greater than or equal to the end time.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	56	Q,-,-,-	**Meaning:** Severe error - An empty SMF record type and subtype list (*smf_record_type_list* and *smf_record_type_info* = INCLUDE) was specified for the ERBDSQRY service.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	57	Q,-,-,-	**Meaning:** Severe error - An empty SMF system name list (*smf_system_name_list* and *smf_system_name_info* = INCLUDE) was specified for the ERBDSQRY service.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.

Table 11. RMF Sysplex Data Services Return and Reason Codes (SMF Services) (continued)

Return Code	Reason Code	Service Routine	Meaning / Action
16	58	-,R,-,-	**Meaning:** Severe error - An empty record token list (*rmf_record_token_list*) was specified for the ERBDSREC service.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	60	Q,R,2,3	**Meaning:** Severe error - RMF could not access one or more of the parameters.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	61	Q,R,2,3	**Meaning:** Severe error - RMF could not access the answer area via the specified ALET (*answer_area_alet*).
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	70	Q,R,2,3	**Meaning:** Severe error - The answer area provided by the calling program (*answer_area_addr* and *answer_area_length*) header was too small to contain even the information.
			Action: Examine your program to locate the CALL that caused the error condition. Correct the statements that are wrong, and rerun your program.
16	71	Q,R,-,-	**Meaning:** Severe error - The requested storage could not be allocated.
			Action: Increase the size of the region where the calling program is running.
16	80	Q,R,-,-	**Meaning:** Severe error - The user is not authorized to call the RMF sysplex data services for SMF data (ERBDSQRY, ERBDSREC, ERB2XDGS and ERB3XDRS).
			Action: Contact your local security administrator.
16	81	Q,R,2,3	**Meaning:** Severe error - The calling program is not in task mode.
			Action: Rerun your program in the correct mode.
16	82	Q,R,2,3	**Meaning:** Severe error - The calling program is not enabled.
			Action: Rerun your program in the correct mode.
16	83	Q,R,2,3	**Meaning:** Severe error - The calling program is not unlocked.
			Action: Rerun your program in the correct mode.
16	84	-,-,2,-	**Meaning:** Severe error - The user is not authorized to access Monitor II data.
			Action: Contact your local security administrator.
16	85	-,-,-,3	**Meaning:** Severe error - The user is not authorized to access Monitor III data.
			Action: Contact your local security administrator.
16	90	Q,R,2,3	**Meaning:** Severe error - RMF encountered a severe error when calling the service routine. This may be caused by a terminating RMF address space.
			Action: Restart RMF and rerun your program.
16	91	-,-,2,3	**Meaning:** Severe error - RMF encountered a severe error when loading the service exit routine. The routine was not found.
			Action: Ensure the exit routine is properly installed on all systems the request is directed to. Rerun your program.
16	92	-,-,2,3	**Meaning:** Severe error - RMF recognized a severe error when executing the service exit routine. The exit completion code is provided in the answer area returned by the service.
			Action: Correct the exit routine problems and rerun your program.
20	0	Q,R,2,3	**Meaning:** Unrecoverable error - An unrecoverable RMF error was encountered during the processing of the requested service. This situation is normally accompanied by error messages sent to the system console and/or a dump.
			Action: Notify your system programmer.

End of Programming Interface information

Layout of RMF callable services answer area

Programming Interface information

When the RMF Sysplex Data Services complete successfully and return control to your program, the answer area contains a common header and one or more data sections.

Layout of common answer area header

The layout for the common callable service answer area header is:

NAM	VER	LEN	TLN
PLX		SOF	SLN
SNO	DOF	DLN	DNO
SNM1		SID1	RMF1
SNM2		SID2	RMF2
…		…	…

where:

NAM Four-character acronym of the common header as follows:
- 'DSQA' for ERBDSQRY/ERBDSQ64
- 'DSRA' for ERBDSREC/ERBDSR64
- 'XDGH' for ERB2XDGS/ERB2XD64
- 'XDRH' for ERB3XDRS/ERB3XD64

VER Version of the common header (initially set to 1).

LEN Total length of the returned data.

TLN Total length of the answer area needed to contain all the requested data.

PLX Name of the sysplex on which the calling application is running.

SOF Offset from the header to the first system list entry SNM.

SLN Length of one system list entry (SNM,SID,RMF).

SNO Number of system list entries (SNM,SID,RMF).

DOF Offset from the header to the first data section. For the detailed layout, refer to the individual data section explanations.

DLN Length of one data section. For a variable length data section, this field is zero. In this case, the length is stored in the individual data section header.

DNO Number of returned data sections.

system list
contains one entry per system in the sysplex:

SNMn
8-character system name

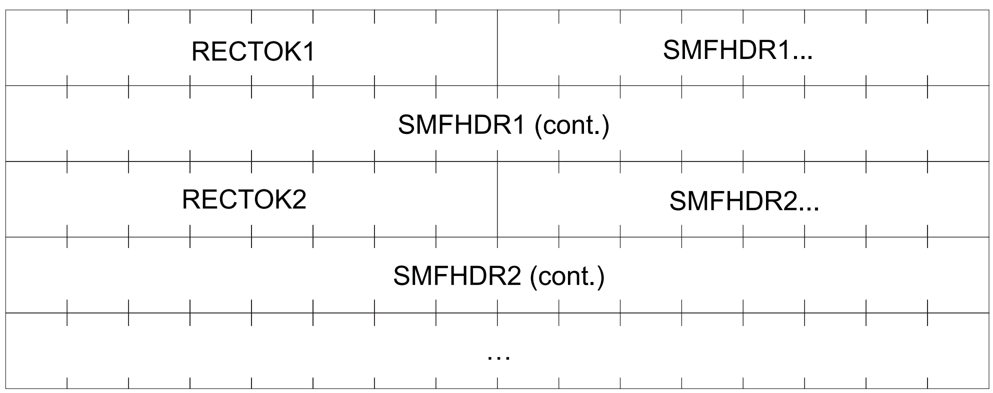
SIDn 4-character SMF system ID. If RMF is not active on this system, this field contains hex zeros.

RMFn 32-bit RMF status indicator, in which:
- Bit 0 (high-order bit) indicates the status of the RMF address space on this system ('1'B = active)
- Bit 1 indicates the status of the RMF Data Buffer for SMF data on this system ('1'B = active)
- Bit 2 indicates the status of the RMF Monitor III address space on this system ('1'B = active)
- Bits 3 to 31 are reserved

ERBDSQRY/ERBDSQ64 data section layout

When ERBDSQRY completes successfully and returns control to your program, the answer area contains the common header plus one directory entry for each SMF record. The directory entry contains a record token created by ERBDSQRY, which may be used for a subsequent call to ERBDSREC to request the actual SMF record itself, and the SMF record header.

The complete layout for the answer area directory entry for *request_type* = **SMF** is:



where:

RECTOKENn Record token provided by ERBDSQRY to be used on subsequent calls to ERBDSREC.

SMFHDRn SMF record header (24 bytes) as described in *z/OS MVS System Management Facilities (SMF)*. For SMF record types without subtypes, which have a header only 18 bytes long, bytes 19 to 24 contain hex zeros.

Name	Length	Format	Description.
SMFxxLEN	2	Integer	SMF record length
SMFxxSEG	2	Integer	SMF segment descriptor
SMFxxFLG	1	Binary	SMF system indicator
SMFxxRTY	1	Integer	SMF record type
SMFxxTME	4	Integer	SMF record time (1/100 sec)
SMFxxDTE	4	0CYYDDDF	SMF record date

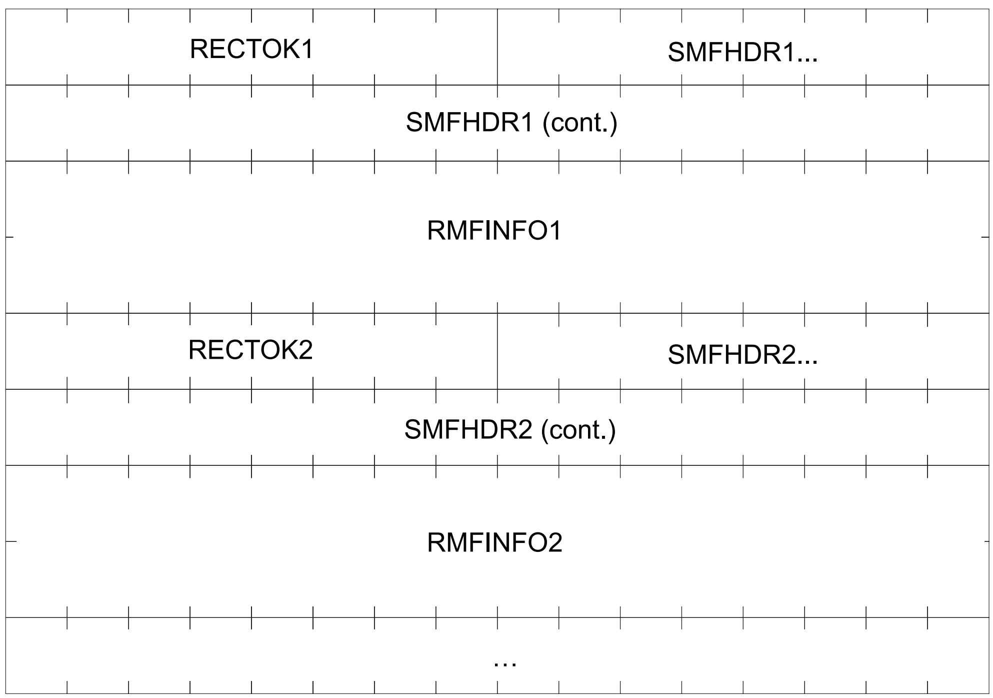
Answer area

Name	Length	Format	Description.
SMFxxSID	4	Char	SMF system id
SMFxxSSI	4	Char	SMF subsystem id
SMFxxSTY	2	Integer	SMF record subtype

For *request_type* = **SMF**, the directory entries are sorted by:

1. **SMFxxDTE**: SMF record date
2. **SMFxxTME**: SMF record time
3. **SMFxxRTY**: SMF record type
4. **SMFxxSTY**: SMF record subtype
5. **SMFxxSID**: SMF record system ID

For *request_type* = **RMF** only, each directory entry contains **additional** information from the RMF product section of the SMF record. The layout for *request_type* = *RMF* is:



where:

RMFINFOn For *request_type* = **RMF**, this field contains 32 bytes of additional information from the RMF product section of the SMF record:

Name	Length	Format	Description.
SMFxxDAT	4	0CYYDDDF	RMF actual interval start date
SMFxxIST	4	0HHMMSSF	RMF actual interval start time
SMFxxINT	4	MMSSTTTF	RMF actual interval length
SMFxxOIL	2	Integer	RMF projected interval length (seconds)
SMFxxSYN	2	Integer	RMF synchronization value (seconds)

Name	Length	Format	Description.
SMFxxLGO	8	(STCK)	RMF offset GMT to local time
SMFxxGIE	8	(STCK)	RMF projected interval end (GMT)

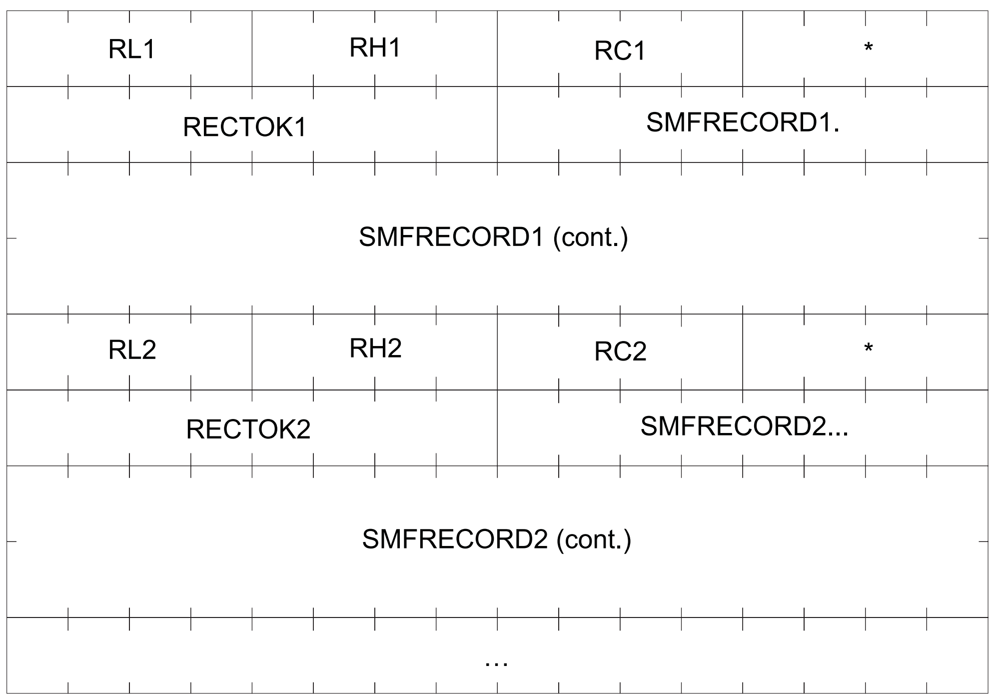
For *request_type* = **RMF**, the directory entries are sorted by:

1. **SMFxxDAT**: RMF interval start date
2. **SMFxxIST**: RMF interval start time
3. **SMFxxRTY**: SMF record type
4. **SMFxxSTY**: SMF record subtype
5. **SMFxxSID**: SMF record system ID

ERBDSREC/ERBDSR64 data section layout

When ERBDSREC returns control to your program after the service was completed successfully, the answer area contains the common header and one entry for each requested SMF record. The entries appear in the order of the request, which is identical to the order of the tokens in the record token list. The entry for each record contains a data header, which is provided by ERBDSREC, and the SMF record itself.

The complete layout of the data section is as follows:



where:

RLn
: Length of this SMF record data entry, including the data header

RHn
: Length of this SMF record data header

RCn
: Return code for the request of this SMF record:

  0 Data returned. SMF record data follows this data header

	4	Data not returned. Timeout occurred before the record was received from the remote system
	8	Data not returned. The record token does not correspond to an existing SMF record in the sysplex

RECTOKn Record token for this SMF record (copied from input parameter)

SMFRECORDn
SMF record

ERB2XDGS/ERB2XD64 data section layout

When ERB2XDGS returns control to your program after the service was completed successfully, the answer area contains the common header and one or more data sections. Each data section contains a data header followed by the Monitor II data itself.

The layout of the data header is

DEL	HDL	RTN	RSN
CPU	PRT	DRC	
…	SRM	SID	*
ZAP	ZIP		

where:

DEL Length of this data section

HDL Length of this data header

RTN Data Retrieval return code

RSN Data Retrieval reason code

CPU System CPU utilization of standard CPs (if Monitor I CPU gathering is not active, this field has the value '-1')

PRT System Paging Rate

DRC Data Reduction exit completion code, if the exit ended abnormally. The completion is in the format TCCCRRRRRRRR, where:
- T is 'S' or 'U' for a system or user completion code, respectively
- CCC is the hexadecimal completion code. The highest possible user completion code is x'FFF'.
- RRRRRRRR is the hexadecimal reason code associated with the completion code.

SRM MVS view of CPU utilization of standard CPs if Monitor I CPU gathering is active, otherwise the SRM view of the CPU utilization (CCVUTILP).

SID SMF system ID.

ZAP System CPU utilization of zAAPs (if Monitor I CPU gathering is not active, this field has the value '-1')

ZIP System CPU utilization of zIIPs (if Monitor I CPU gathering is not active, this field has the value '-1')

***** Reserved.

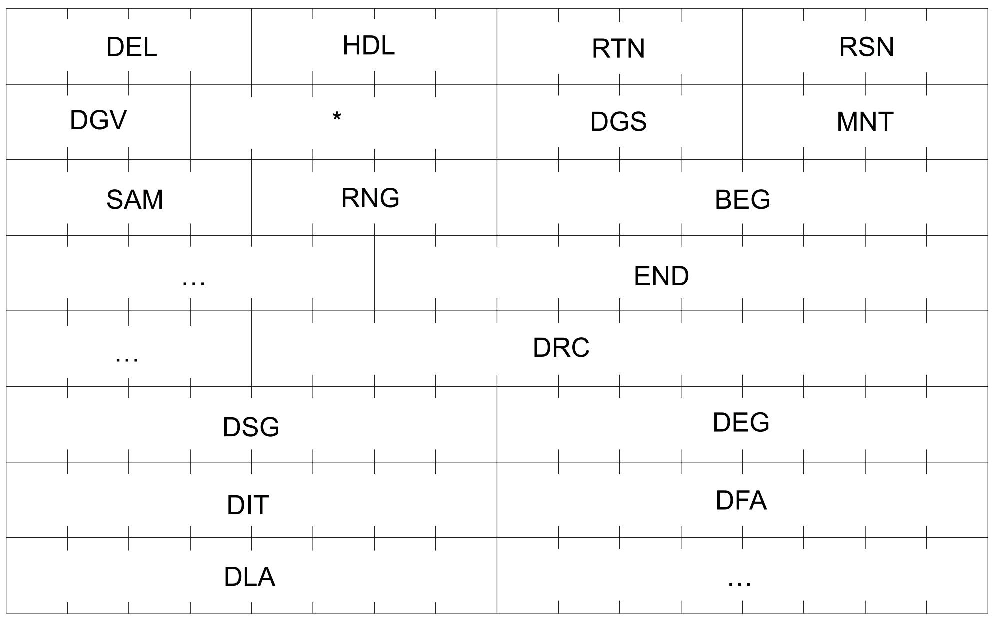
Each data section contains the data header described above, followed by the data provided by the data reduction exit routine.

ERB3XDRS/ERB3XD64 data section layout

When ERB3XDRS returns control to your program after the service has completed successfully, the answer area contains the common header and one or more data sections. Each data section contains a data header followed by the Monitor III data itself. The layout of the data section is as follows:

- One or more set-of-samples. The layout of the uncompressed set-of-samples is described in "Data gatherer sample structure" on page 158.

The layout of the data header is



where:

DEL Length of this data section

HDL Length of this data header

RTN Data Retrieval return code

RSN Data Retrieval reason code

DGV Data gatherer version in the format 'VRM'.

DGS System name of the system on which the data gatherer is running

MNT Data gatherer MINTIME option

SAM Actual number of samples in the returned data

RNG Actual range length in seconds

BEG Actual range start time in the format YYYYMMDDHHMMSS.

END Actual range end time in the format YYYYMMDDHHMMSS.

DRC Data Reduction exit completion code, if the exit ended abnormally The completion code is in the format TCCCRRRRRRRR, where:
- T is 'S' or 'U' for a system or a user completion code, respectively
- CCC is the hexadecimal completion code
- RRRRRRRR is the hexadecimal reason code associated with the completion code

The following fields will be filled with Monitor III data statistics for certain warning and error conditions.

For return code 8 or 12 and reason code 8 or 9:

DSG Start time of a time gap in the Monitor III data in store clock format

DEG End time of a time gap in the Monitor III data in store clock format

For return code 8 or 12 and reason code 15:

DIT IPL time of the system in store clock format

For return code 12 and reason code 16:

DFA Start time of the Monitor III data that is available for reporting on this system in store clock format

DLA End time of the Monitor III data that is available for reporting on this system in store clock format

* Reserved

Note: The data header length field contains 120 instead of 80 if the additional data statistics are present. If the systems in the sysplex have a different RMF service level, both data header formats may appear in the same ERB3XDRS answer area.

Each data section contains the data header described above, followed by the data provided by the data reduction exit routine.

End of Programming Interface information

Chapter 3. Accessing performance data using the RMF Distributed Data Server

With the two flavours of the Distributed Data Server (GPMSERVE and GPM4CIM, aka RMF XP), RMF offers a solution to monitor the performance of systems in heterogeneous environments. Application programs which want to retrieve performance data can use the HTTP API of the Distributed Data Server (DDS). The DDS returns the requested performance data as a structured XML document.

- For systems in a z/OS environment, the DDS GPMSERVE component gathers data from the RMF instances running on the sysplex members. An application program can request selected performance metrics or complete reports from the DDS. Thus, exploiters of this HTTP API have instant access to a great variety of z/OS performance metrics, including short-term information as well as long-term historical data.
- The HTTP API of the DDS can also serve requests for AIX and Linux performance data, which are directed against an active GPM4CIM instance. RMF XP supports the following operating systems:
  - AIX on System p
  - Linux on System x
  - Linux on System z

  GPM4CIM is the core component of RMF XP. For detailed information about RMF XP, refer to the *z/OS RMF User's Guide*.

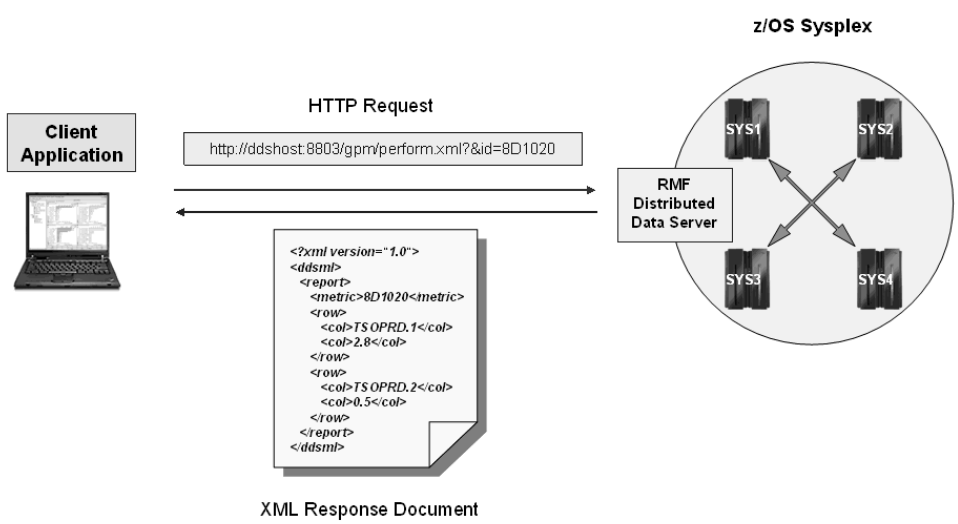


Figure 8. Example: How to use the DDS HTTP API in a z/OS environment

Figure 8 illustrates how to use the DDS HTTP API in a z/OS environment, where requests must be sent to the GPMSERVE component of the DDS. Likewise, in a heterogeneous AIX/Linux environment, you send your HTTP request to the corresponding GPM4CIM instance of the DDS. Each instance of GPMSERVE or GPM4CIM uses a unique port to listen for incoming requests. The returned XML documents have the same syntax for both Distributed Data Server components.

This information unit describes the format and usage of DDS requests as well as the format of the returned XML documents in the following topics:

- “How to specify HTTP requests to the DDS for performance data”
- “How to interpret an XML document returned by the DDS” on page 59

How to specify HTTP requests to the DDS for performance data

You can specify a variety of requests for different purposes. For example, you can request:

- the children of a resource
- a metric value for a resource
- a list of associated metrics for a resource
- a list of associated details for a resource
- a selection of Monitor III and Postprocessor reports

The required information how to specify such requests is contained in the following topics:

- “Understanding the underlying resource models”
- “Structure of DDS requests” on page 48
- “Description and purpose of parameters” on page 49
- “How to specify different types of requests” on page 56

Understanding the underlying resource models

It is useful to understand the concept of the used resources and their associated metrics, because most requests are specified against a resource from which you want to retrieve performance data.

The hierarchies of resources in the supported environments are illustrated in the following topics:

- “The z/OS RMF Monitor III resource model” on page 45
- “The AIX resource model” on page 46
- “The Linux on System x resource model” on page 46
- “The Linux on System z resource model” on page 47

The z/OS RMF Monitor III resource model

The z/OS RMF Monitor III resource model represents a composition of resources with the SYSPLEX as top-level resource. All other resources are children or grand-children of the SYSPLEX, for example, ***PROCESSOR*** and ***STORAGE*** resources are children of an ***MVS Image*** and grandchildren of a ***SYSPLEX***. The entire hierarchical model looks as shown in Figure 9.

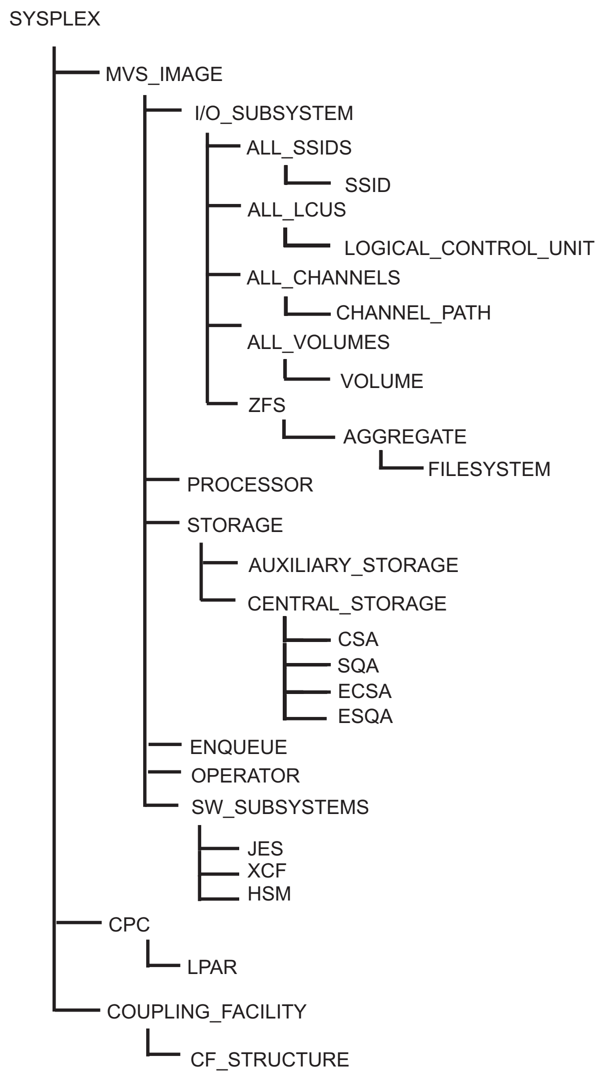


Figure 9. The z/OS RMF Monitor III resource model

The AIX resource model

The AIX resource model represents a composition of resources with an AIX_SYSTEM_COMPLEX as top-level resource. An AIX_SYSTEM_COMPLEX resource denotes a complex of distributed systems running AIX, where the performance data is gathered by a CIM server. All other resources are children or grand-children of the AIX_SYSTEM_COMPLEX. The entire hierarchical model looks as shown in Figure 10.

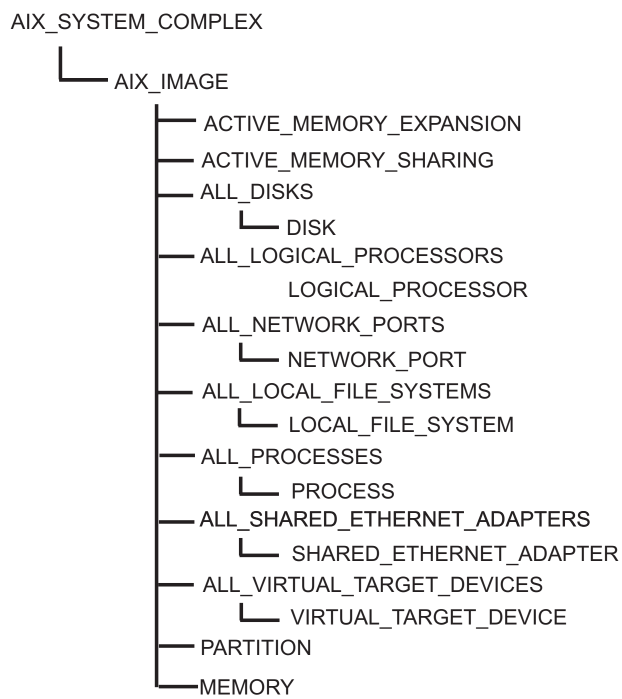


Figure 10. The AIX resource model

The Linux on System x resource model

The Linux on System x resource model represents a composition of resources with an XLINUX_SYSTEM_COMPLEX as top-level resource. An XLINUX_SYSTEM_COMPLEX resource denotes a complex of distributed systems running Linux on System x, where the performance data is gathered by a CIM server. All other resources are children or grand-children of the XLINUX_SYSTEM_COMPLEX. The entire hierarchical model looks as shown in Figure 11 on page 47.

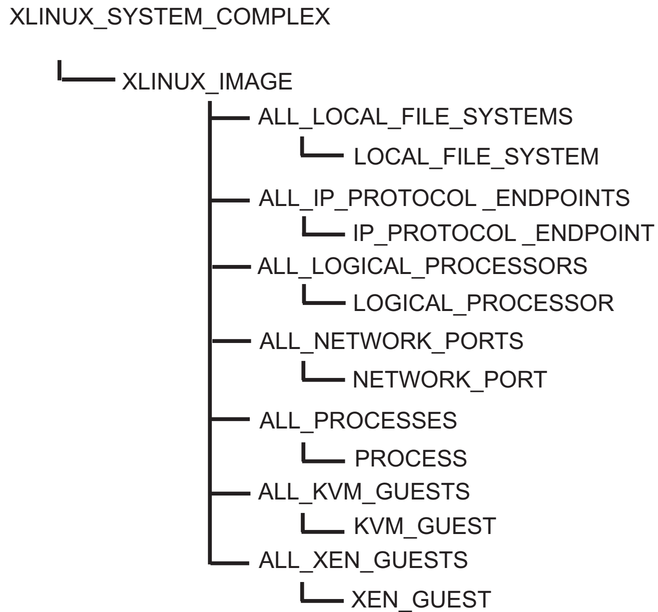


Figure 11. The Linux on System x resource model

The Linux on System z resource model

The Linux on System z resource model represents a composition of resources with a ZLINUX_SYSTEM_COMPLEX as top-level resource. A ZLINUX_SYSTEM_COMPLEX resource denotes a complex of distributed systems running Linux on System z, where the performance data is gathered by a CIM server. All other resources are children or grand-children of the ZLINUX_SYSTEM_COMPLEX. The entire hierarchical model looks as shown in Figure 12.

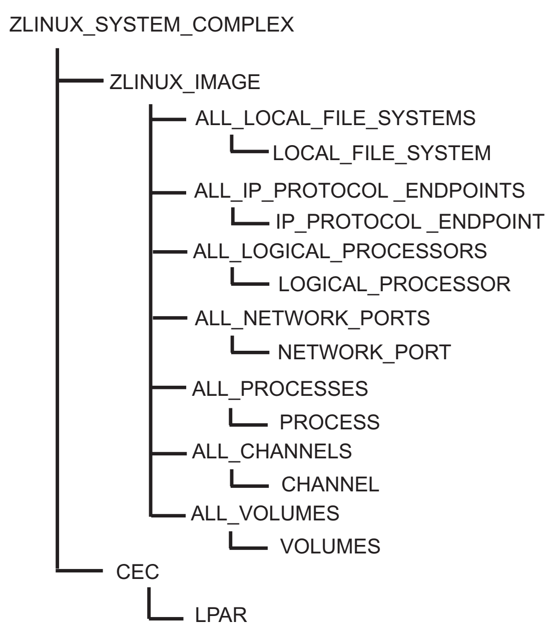


Figure 12. The Linux on System z resource model

Structure of DDS requests

To request performance data in a z/OS environment, an application must send an HTTP request to the DDS server located on the monitored z/OS sysplex.

To request cross platform performance data from a Linux or AIX system, the HTTP request must be sent to the DDS server with the monitored Linux or AIX system defined in its configuration file.

Topic "Description and purpose of parameters" on page 49 first explains the set of request parameters, used in the various request types. Then, "How to specify different types of requests" on page 56 presents detailed information about the purpose of the request types and how to specify them.

Here is an example request for a certain performance metric for a specified resource: it requests the response time (denoted by the metric ID ***8D10F0***) of volume ***SYSLIB*** of system ***SYSA***:

```
http://ddshost:8803/gpm/perform.xml?resource=SYSA,SYSLIB,VOLUME&id=8D10F0
```

An example request for the Postprocessor CPU and CRYPTO reports looks similar to the following:

```
http://ddshost:8803/gpm/rmfpp.xml?reports=CPU,CRYPTO&date=20090801,20090804
```

General request syntax:

```
http://<ddshost>:<ddsport>/gpm/<filename>?<parm_1>=<value_1>&...&<parm_n>=<value_n>
```

The parameters have the following meanings:

ddshost
is the IP address or the symbolic name of the DDS server.

Example:

```
ddshost
```

ddsport
is the port number of the DDS server (GPMSERVE or GPM4CIM).

Example:

```
8803
```

filename
is the filename of the XML document you want to retrieve, followed by the extension `.xml`. It determines the request type and the returned XML document type. The valid filenames are described in "How to specify different types of requests" on page 56.

Examples:

```
perform.xml
rmfpp.xml
```

<parm_1>=<value_1>&...&<parm_n>=<value_n>
is the query string within the request. It is composed of multiple parameter/value pairs, separated by **&** characters, that precisely specify the requested information. Available parameters are described in "Description and purpose of parameters" on page 49.

Description and purpose of parameters

Table 12 shows an overview of parameters which you can specify in the query string of the request. The remainder of this topic describes the parameters in detail.

Table 12. Request parameters

Parameter	**Purpose**
Parameters for Monitor III and CIM requests	
resource	Monitor III resource identifier
id	metric identifier associated with the resource
range	start and end time of data interval
filter	filter string
workscope	workscope specification
name	name of list element
listtype	type of list element
Parameters for Postprocessor requests	
reports	list of Postprocessor report names
date	start and end date of the reporting period for the requested Postprocessor report(s)
duration	interval length for the requested Postprocessor duration report(s)
timeofday	start and end time of the reporting period for each day in the reporting period
sysid	system name for single system reports
overview	list of control statements for the Overview report
timeout	timeout period in seconds for the completion of Postprocessor jobs

The *resource* parameter

resource=[ulq],[resource_name],resource_type

This parameter describes the resource for which information is requested. The resource parameter is composed of three parts:
- upper level qualifier (ULQ)
- resource name
- resource type

You can see the available resource types in the syntax required for the request string in Figure 9 on page 45.

An ULQ is needed for the resource parameter, because resources with the same name can exist multiple times in a sysplex, for example volumes or channels. For most of the resources, the ULQ is the name of the associated z/OS system.

For the sysplex resource, the ULQ can be omitted. In such a case, the resource specification starts with a comma. For unique resources like the PROCESSOR resource in an MVS IMAGE, you can either omit the resource name or you can assign an asterisk as the resource name.

Examples for Monitor III resources:

- An MVS image named ***SYSA*** in a sysplex is represented as `resource=,SYSA,MVS_IMAGE`. That is, in this case the upper level qualifier can be omitted, since the system name is unique in a sysplex.
- An instance of resource type CHANNEL_PATH with ID ***0F*** in MVS_IMAGE ***SYSA*** is represented as `resource=SYSA,0F,CHANNEL_PATH`. Here, all three parts of the resource identifier are required, because channel paths are not unique in a sysplex.
- An instance of resource type PROCESSOR, which is unique in MVS_IMAGE ***SYSA*** is represented as `resource=SYSA,,PROCESSOR` or `resource=SYSA,*,PROCESSOR`.

Examples for Linux/AIX resources: (resource names are case-sensitive)

- A Linux on System z image named ***LINZRMF5*** is represented as `resource=,LINZRMF5,ZLINUX_IMAGE`. You can omit the upper level qualifier, as system names are unique in a ZLINUX_SYSTEM_COMPLEX, and also are unique in an XLINUX_SYSTEM_COMPLEX and in an AIX_SYSTEM_COMPLEX, too.
- An AIX image named ***p6rmf2*** is represented as `resource=,p6rmf2,AIX_IMAGE`.
- A process on a Linux image named ***LINXRMF5*** with resource name ***102*** is represented as `resource=LINXRMF5,102,PROCESS`. Here, all three parts of the resource identifier are required, because processes are not unique in Linux/AIX complexes.

  `resource=p6rmf2,cimlistener[7209212],PROCESS` is an example of a process on an AIX image.
- An instance of resource type LOGICAL_PROCESSOR with resource name ***CPU0*** on system ***P6RMF1*** is represented as `resource=P6RMF1,CPU0,LOGICAL_PROCESSOR`. This notation is valid for either Linux or AIX.

The *id* parameter

id=metric_ID

This parameter identifies the Monitor III metric that you want to request. The metric identifier must be applicable for the resource given in the same request. You can retrieve a metric ID in the following ways, depending on whether you search for any arbitrary metric or for a metric that is associated with a certain resource:

1. To find any arbitrary metric ID, search the file `http://<ddshost>:<ddsport>/gpm/index.xml` for the desired metric. For example, searching for the metric ***# active users*** would return the desired information:

   ```
   <metric id="8D0620">
     <description># active users</description>
   ...
   ...
   ```
2. To find a metric ID that is associated with a certain resource (in our example for a resource of type `MVS_IMAGE`), specify a listmetrics request similar to the following one:

   ```
   http://<ddshost>:<ddsport>/gpm/listmetrics.xml?resource=,<system_name>,MVS_IMAGE
   ```

   The returned metric list contains the desired information in the same format as shown for alternative 1.

The *range* parameter

range=YYYYMMDDhhmmss[,YYYYMMDDhhmmss]

This parameter specifies the time interval for the requested performance data. Start and end times are specified as a string of the form `YYYYMMDDhhmmss`, provided in local time of the monitored system. If you omit the end time, the defined Monitor III gatherer interval (MINTIME) is used to determine the end time.

Example:

```
RANGE=20090725100000,20090725110000
```

specifies the start time as 10:00 AM on 07/25/2009 and the end time as 11:00 AM on the same day.

The *filter* parameter

filter=list_of_filter_criteria

You can focus on the data of your interest by adding a filter specification when requesting a list of values. You can use filters to specify the following:
- one or more name patterns to be matched against the names in the list
- a lower and upper bound to be compared to the values in a list
- a maximum list length with an indicator to select the instances with either the highest or the lowest values
- a sorting order for either the names or the values of the list (ascending or descending).

You can separate multiple filter criteria by a semicolon (";").

The following keywords are available for filters:

PAT=<expression>
Specifies one or more patterns which must match the name part of a list element.

LB=<double> Specifies a lower bound value. That is, only list elements with values higher than the given lower bound are returned.

UB=<double> Specifies an upper bound value. That is, only list elements with values lower than the specified upper bound are returned.

HI=<integer> Only the highest <integer> list elements are returned (mutually exclusive with LO). The default is 20.

LO=<integer> Only the lowest <integer> list elements are returned (mutually exclusive with HI).

ORD=<xx> Sort the list of name/value pairs by their names in ascending (NA) or descending (ND) order (<xx>=NA or ND), or by their values in ascending (VA) or descending (VD) order. The default is VD. If you do not want to have any sort order, specify ORD=NN.

If you do not explicitly specify a filter, the default settings are `&filter=HI=20;ORD=VD` which returns the top 20 values, sorted by value, in descending order (VD).

Examples:

&filter=PAT=*CICS*|*SH*	only instances containing the name patterns CICS, SH, or both are returned
&filter=LB=10.5	only instances with a value higher than 10.5 are returned
&filter=UB=12.8	only instances with a value lower than 12.8 are returned

&filter=HI=5	only the instances containing the five highest values are returned
&filter=LO=5	only the instances containing the five lowest values are returned
&filter=ORD=NA	specifies that the returned instances should be sorted by name in ascending order

The *workscope* parameter

workscope=[ulq],name,type

Use this parameter to qualify a request for performance data in more detail with regard to address spaces and WLM entities. Workscopes can be applied to single valued metrics as well as to list valued metrics. For example,
- for the metric *performance index*, the workscope parameter denotes the associated service class period
- for the metric *% workflow by job*, you can use this parameter to focus on jobs that belong to a certain service class.

The workscope parameter consists of three parts:
- an upper level qualifier which may be blank or which specifies the name of a WLM service class, if the workscope type is a WLM service class period
- a workscope name (for example, job name or report class name) or a service class period
- a workscope type

  Available workscope types:

  G global (no workscope required)
  W WLM workload
  S WLM service class
  P WLM service class period
  R WLM report class
  J job

Examples:

&workscope=,BATCH,W	focus on workload BATCH
&workscope=HOTTSO,3,P	focus on period 3 of service class HOTTSO
&workscope=,CICSPRD,R	focus on report class CICSPRD
&workscope=,CATALOG,J	focus on job catalog

Example request strings with a workscope parameter:

Example 1 (single valued metric): Retrieve the performance index (ID = 8D1000) for the first period of service class BATCHMED:

```
http://ddshost:8803/gpm/perform.xml?resource=,MVSPLEX,SYSPLEX
                                            &id=8D1000&workscope=BATCHMED,1,P
```

Example 2 (list valued metric): Retrieve the workflow by job (ID = 8D0560) for all jobs running in service class HOTTSO:

```
http://ddshost:8803/gpm/perform.xml?resource=,SYSA,MVS_IMAGE
                                            &id=8D0560&workscope=,HOTTSO,S
```

The *name* parameter

name=resource_name

This parameter is required in a ***detailsname*** request, which returns the attributes of a resource. In such a request, the ***name*** parameter specifies the name of a resource for which the attributes are requested. For example, you can retrieve the attributes of a volume with a specified name as shown in "How to specify a ***detailsname*** request" on page 57.

The *listtype* parameter

listtype=type

This parameter is required for the following request types:

- in a ***detailsname*** type request together with the ***name*** parameter. It specifies the list type of the resource for which details are requested. For example, if you want to know attributes of a volume, you need to specify ***listtype=V***.
- in a ***filterinst*** type request. In such a request, the list type denotes the resource type of the requested filter instances.

  For example, you use the ***listtype*** parameter to retrieve either volumes (listtype=V) or channels (listtype=C) as filter instances for the sysplex resource.

Available z/OS list types:

A partition
B job (only for SYSPLEX resource)
C channel path
D data set
E enclave
F coupling facility
G zFS aggregate
I SSID
J job
K WLM report class period
L LCU
M MVS image
O USS pid and job
P WLM service class period
Q storage group
R WLM report class
S WLM service class
T CF structure
U channel path and CU
V volume
W WLM workload
X CPC
Y zFS file system

Available AIX list types:

A shared ethernet adapter
D disk
F file system
I AIX image
L logical processor
N network port
P process
V virtual target device

Available Linux on System x list types:

E IP protocol endpoint

F	local file system
I	Linux image
K	KVM domain
L	logical processor
N	network port
P	process
X	XEN domain

Available Linux on System z list types:

C	CEC
E	IP protocol endpoint
F	local file system
H	channel
I	Linux image
L	logical processor
N	network port
P	process
R	LPAR
V	volume

The *reports* parameter

reports=report_name[(options)][, report_name[(options)], ... report_name[(options)]]

This parameter enumerates one or more identifiers of Postprocessor reports to be returned by the request. You may define options for applicable reports, as described in the *z/OS RMF User's Guide* in topic ***Long-term reporting with the Postprocessor***.

Examples:

- Get detailed data for service class TSOPROD by requesting the Service Class report:
  `reports=WLMGL(SCLASS(TSOPROD))`
- Create a Workload Activity report and assume that all CICS applications run in the three workload groups CICSPROD, CICSTEST, and CICSADMN. Get the Workload Group report for all groups:
  `reports=WLMGL(WGROUP(CICS*))`
- Request a list of Postprocessor reports, with the WLMGL report with suboptions:
  `reports=CPU,CRYPTO,WLMGL(SCPER(STCLOW))`

Notes:

1. The enumerated report identifiers can be enclosed in double quotes.
2. You cannot use the ***reports*** parameter and the ***overview*** parameter in the same request. That is, you can either request one or more standard Postprocessor reports or one Overview report (see “The ***overview*** parameter” on page 55).

For a complete list of available Postprocessor reports in XML format refer to the *z/OS RMF User's Guide*.

The *date* parameter

date=<start-date>,<end-date>

This parameter specifies the start and end date of the reporting period for all Postprocessor reports in the format `yyyymmddhh` or `yyddd`.

Examples:

```
date=20101125,20101126
date=10256,10257
```

The *duration* parameter

duration=<interval-length>

This parameter specifies that the Postprocessor is to generate duration reports and indicates the length of the duration interval in the format ***hhmm***. The minimum value is `0000` which is corrected by the Postprocessor to the interval length that is found in the data being processed. The maximum value is `9960` which is equivalent to 100 hours.

Example:

```
duration=1200
```

The *timeofday* parameter

timeofday=<start-time>,<end-time>

This parameter specifies the start and end time of the reporting period for each day in the reporting period in the format `hhmm`.

Example:

```
timeofday=0800,1600
```

The *sysid* parameter

sysid=<cccc>

This parameter identifies the single system for which the reports are to be generated. It is ignored for sysplex reports.

Example:

```
sysid=SYSA
```

The *overview* parameter

overview=(<statement_1>),(<statement_2>)...(<statement_n>)

where <statement_n> is

```
control_statement_name(condition_name(qualifier))[,SYSTEMS | ,NOSYSTEMS]
```

This parameter contains a list of control statements for the Overview report, equivalent to the OVW control statements as described in the *z/OS RMF User's Guide*. The maximum number of control statements is 253.

Example:

The request parameter

```
overview=(DATA01(CADSTG(SSID(0600),DEVN(06F3)))),
(DB2PRD(CADRT(DEVN(0722),SSID(0700)))),
(RHT0050(CASRHT(SSID(0050))))
```

represents the following overview control statements:

```
OVW(DATA01(CADSTG(SSID(0600),DEVN(06F3))))
OVW(DB2PRD(CADRT(SSID(0700),DEVN(0722))))
OVW(RHT0050(CASRHT(SSID(0050))))
```

The *timeout* parameter

timeout=<wait-time>

This parameter specifies the timeout period in seconds, that the DDS should wait for Postprocessor jobs to complete. The valid range is from 0 to 3600 seconds. The default value is 300 seconds.

Example:

```
timeout=1200
```

How to specify different types of requests

This topic explains the purpose of the available types of requests and describes how to specify each request type. A request type is determined by the specified value of the ***<filename>*** parameter.

There is one subtopic for each available request type, presenting an example request and listing the required and optional parameters. The resulting XML documents are documented in “How to interpret an XML document returned by the DDS” on page 59.

Table 13 lists all filename specifications which are valid in a request string to a z/OS system and also indicates which filenames can be specified for Linux or AIX.

Table 13. Valid filename specifications

Filename	see page	Purpose	XML document type	valid for AIX and Linux
contained.xml	56	list of child resources	<contained-resources-list>	yes
details.xml	57	attributes of a resource	<attribute-list>	no
detailsname.xml	57	attributes of a resource	<attribute-list>	no
filterinst.xml	57	list of filter instances	<filter-instances-list>	yes
index.xml	57	list of metrics for all resources	<metric-list>	yes
listmetrics.xml	57	list of metrics for one resource	<metric-list>	yes
perform.xml	58	performance data	<report>	yes
<report_name>.xml	58	Monitor III report	<report>	no
rmfpp.xml	59	one or more Postprocessor reports	<postprocessor>	no
root.xml	58	root resource	<contained-resources-list>	yes
workscopes.xml	58	list of workscopes	<workscope-list>	no

How to specify a *contained* request

A request using this filename returns the list of children for the specified resource. The result is an XML file of type <contained-resources-list>.

Example request strings:

```
http://ddshost:8803/gpm/contained.xml?resource=,MVSPLEX,SYSPLEX
http://ddshost:8805/gpm/contained.xml?resource=,,AIX_SYSTEM_COMPLEX
http://ddshost:8805/gpm/contained.xml?resource=,P6RMF1,AIX_IMAGE
```

Required parameter: *resource*

How to specify a *details* request

A request using this filename returns the list of attributes for the selected resource. The result is an XML file of type <attribute-list>.

Example request string:

```
http://ddshost:8803/gpm/details.xml?resource=,SYSA,MVS_IMAGE
```

Required parameter: *resource*

How to specify a *detailsname* request

A request using this filename returns the list of attributes for the single resource designated by the ***name*** parameter. You must also specify the type of the list, for example &listtype=V for volumes. The result is an XML file of type <attribute-list>.

Example request string:

```
http://ddshost:8803/gpm/detailsname.xml?resource=SYSA,*,ALL_VOLUMES&name=SYSLIB
                                                                    &listtype=V
```

Required parameters: *resource, name, listtype*

How to specify a *filterinst* request

A request using this filename returns a list of all possible filter instances with the specified list type for the given resource. The result is an XML file of type <filter-instances-list>.

Example request strings:

```
http://ddshost:8803/gpm/filterinst.xml?resource=,MVSPLEX,SYSPLEX&listtype=C
http://ddshost:8807/gpm/filterinst.xml?resource=,ZLINUXPLEX,ZLINUX_SYSTEM_COMPLEX
                                                                 &listtype=V
```

Required parameters: *resource, listtype*

How to specify an *index* request

A request using this filename returns the list with all resources and associated metrics in the sysplex. It is invoked without any parameters. The result is an XML file of type <metric-list>.

Example request string:

```
http://ddshost:8803/gpm/index.xml
```

How to specify a *listmetrics* request

A request using this filename returns the list of associated metrics for the specified resource type. The result is an XML file of type <metric-list>.

Example request strings:

```
http://ddshost:8803/gpm/listmetrics.xml?resource=,SYSA,MVS_IMAGE
http://ddshost:8805/gpm/listmetrics.xml?resource=P6RMF1,CPU0,LOGICAL_PROCESSOR
```

Required parameter: *resource*

How to specify a *perform* request

A request using this filename returns performance data for the selected metric of the specified resource. The result is an XML file of type <report>.

Example request strings:

```
http://ddshost:8803/gpm/perform.xml?resource=,SYSA,MVS_IMAGE&id=8D0160
http://ddshost:8806/gpm/perform.xml?resource=,XLINSYSA,XLINUX_IMAGE&id=203160
http://ddshost:8807/gpm/perform.xml?resource=ZLINXRMF,PROC0,LOGICAL_PROCESSOR
                                                                  &id=304010
```

Required parameters: *resource*, *id*

Optional parameters: *range*, *filter*, *workscope*

How to specify a *report* request

A request using this filename returns a complete RMF Monitor III report for the specified resource. The result is an XML file of type <report>.

The request string for a Monitor III report contains the report name, like CHANNEL or SYSSUM, preceded by the subdirectory *reports*. For example, if you use CHANNEL.xml in a request, then the XML document for a CHANNEL report is produced.

Example request string:

```
http://ddshost:8803/gpm/reports/CHANNEL.xml?resource=,SYSA,MVS_IMAGE
```

Required parameter: *resource*

- To request a list of available reports with sysplex scope, specify a listmetrics request for the SYSPLEX resource, for example:

  ```
  http://ddshost:8803/gpm/listmetrics.xml?resource=,MVSPLEX,SYSPLEX
  ```
- To request a list of available reports with single system scope, specify a listmetrics request for an MVS_IMAGE resource, for example:

  ```
  http://ddshost:8803/gpm/listmetrics.xml?resource=,SYSA,MVS_IMAGE
  ```

Available reports are listed in the returned document as follows:

```
<metric id="report_name">
<format>report</format>
</metric>
```

How to specify a *root* request

A request using this filename returns the z/OS SYSPLEX, or the respective Linux or AIX SYSTEM_COMPLEX as root resource. It is invoked without any parameters. The result is an XML file of type <contained-resources-list> containing only the root resource.

Example request string:

```
http://ddshost:8803/gpm/root.xml
```

How to specify a *workscopes* request

A request using this filename returns the list of associated workscopes for the specified resource. The result is an XML file of type <workscope-list>.

- A request against a SYSPLEX resource returns all WLM entities in the sysplex.
- A request against an MVS_IMAGE resource returns the active jobs in the system.

Example request string:

```
http://ddshost:8803/gpm/workscopes.xml?resource=,MVSPLEX,SYSPLEX
```

Required parameter: *resource*

How to specify a *postprocessor* request

A request using filename ***rmfpp.xml*** returns either the requested (list of) standard Postprocessor report(s) or an Overview report, depending on the parameters (see "Description and purpose of parameters" on page 49). The result of the request is an XML file of type <postprocessor> containing the requested report(s).

Example request string:

Request for a list of reports containing the ***CPU*** and the ***CRYPTO*** report between the first and fourth of August 2010:

```
http://ddshost:8803/gpm/rmfpp.xml?reports=CPU,CRYPTO&date=20100801,20100804
```

Required parameter: either *reports* or *overview*

Optional parameters: *date*, *duration*, *timeofday*, *sysid*, *timeout*

How to interpret an XML document returned by the DDS

This topic contains all syntax information needed to read the XML documents returned by the DDS. It describes the syntax rules of all XML tags used in the returned documents. From these documents, your application program can extract the requested performance data.

The data types **token**, **NMTOKEN**, **byte**, and **nonNegativeInteger** of the returned values are used as defined in the XML Schema language.

Description of the XML document structure

Each XML document type starts with the <ddsml> tag, followed by the <server>...</server> specification.

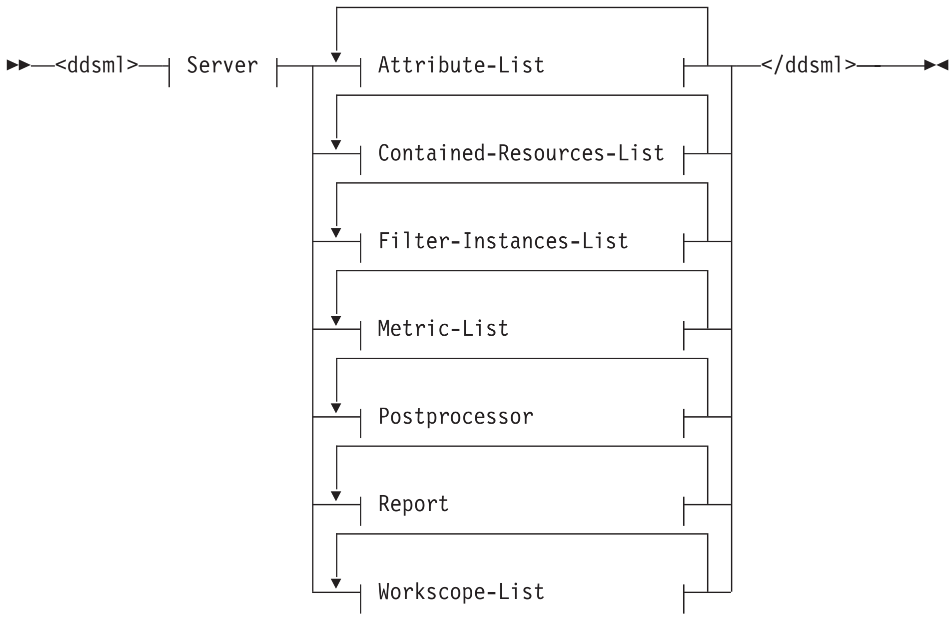
The content which follows the <server>...</server> tags, is enclosed into a pair of corresponding start and end tags, for example:

```
<attribute-list> ... content ... </attribute-list>
```

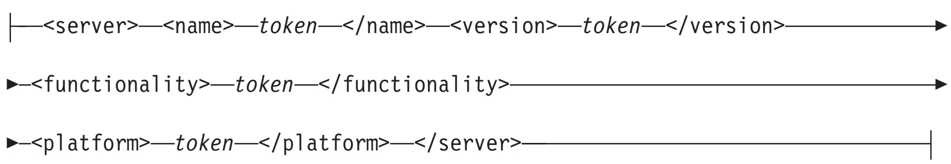
or:

```
<report> ... content ... </report>
```

Syntax: ddsml



Server:



Tag/Fragment	Description	Type
server	characteristics of the DDS server	see syntax diagram
name	name of the DDS server, value is set to ***RMF-DDS-Server***	token
version	operating system release	token
functionality	level number of the DDS server	token
platform	operating system	token
Attribute-List	attributes for a resource	see page 61
Contained-Resources-List	list of child resources	see page 64
Filter-Instances-List	list of filter instances	see page 65
Metric-List	list of metrics for a resource	see page 66
Postprocessor	one or more Postprocessor reports	see page 73
Report	performance data	see page 68
Workscope-List	list of workscopes	see page 72

As mentioned in "How to specify different types of requests" on page 56, the document type is determined by the ***filename*** specification in the request. A specification of the same document type may occur multiple times in one XML document, as shown in the following example, where there are multiple instances of document type ***metric***.

Example:

```
<ddsml xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="/gpm/include/ddsml.xsd">
<server>
<name>RMF-DDS-Server</name>
<version>ZOSV1R13</version>
<functionality>2381</functionality>
<platform>z/OS</platform>
</server>
<metric-list>
<resource>
...
</resource>
<metric id="8D2060">
...
</metric>
...
...
...
<metric id="8D2170">
...
</metric>
</metric-list>
</ddsml>
```

The remainder of this topic describes the syntax of the available document types and presents an example document for each type.

Attribute-List element

The <attribute-list> tag lists a resource and its attributes.

Syntax: Attribute-List

```
                                       ┌──────────────────┐
                                       ▼                  │
►►──<attribute-list>──┤ Resource ├───────┤ Attribute ├──┴──────────────►

►──┬─────────────┬──</attribute-list>─────────────────────────────►◄
   └┤ Message ├──┘
```

Tag/Fragment	Description	Type
Resource	description of a resource	see page 62
Attribute	list of attributes	see page 63
Message	error message or warning	see page 63

Example XML code:

```
<attribute-list>
<resource>
...
</resource>
<attribute id="37">
...
</attribute>
...
<attribute id="78">
...
</attribute>
</attribute-list>
```

Resource

The <resource> tag specifies a resource in detail.

Syntax: Resource

```
►►──<resource>──<reslabel>──token──</reslabel>──────────────────────────────►

►─┬────────────────────────────┬──┬──────────────────────────────┬─────────►
  └─<resulq>──token──</resulq>─┘  └─<resname>──token──</resname>─┘

►─<restype>──token──</restype>──┬──────────────────────────────────┬────────►
                                └─<reslabelurl>──token──</reslabelurl>─┘

►─┬──────────────────────────────────────────┬──┬────────────────────────┬─►
  └─<expandable>──NMTOKEN──</expandable>─────┘  └─<icon>──token──</icon>─┘

►─┬──────────────────────────────────────────┬──────────────────────────────►
  └─<attributes>──NMTOKEN──</attributes>─────┘

►─┬──────────────────────────────┬──</resource>───────────────────────────►◄
  └─<helpurl>──token──</helpurl>─┘
```

Tag/Fragment	Description	Type
reslabel	description of resource	token
resulq	upper level qualifier of the resource	token
resname	resource name	token
restype	resource type	token
reslabelurl	resource label for use in an URL without blanks	token
expandable	denotes whether resource itself contains other resources; value in {YES, NO}	NMTOKEN
icon	name of icon image for this resource	token
attributes	denotes whether attributes may be queried for this resource; value in {YES, NO}	NMTOKEN
helpurl	URL of help description	token

Example XML code:

```
<resource>
<reslabel>,CF01,COUPLING_FACILITY</reslabel>
<resname>CF01</resname>
<restype>COUPLING_FACILITY</restype>
<reslabelurl>,CF01,COUPLING_FACILITY</reslabelurl>
<expandable>YES</expandable>
<icon>rmfcf.gif</icon>
<attributes>YES</attributes>
</resource>
```

Attribute

The <attribute> tag specifies the name and value of each attribute of a resource.

Syntax: Attribute

```
►►──<attribute id="token">──<description>──token──</description>──────────►

►─<value>──token──</value>──</attribute>──────────────────────────────────►◄
```

Tag/Fragment	Description	Type
id	unique attribute ID	token
description	description of a single attribute	token
value	value of this attribute	token

Example XML code:

```
<attribute id="1">
<description>Processor type</description>
<value>2064</value>
</attribute>
<attribute id="16">
<description>unit</description>
<value>3390-9</value>
</attribute>
```

Message

The <message> tag specifies an error message or warning.

Syntax: Message

```
►►──<message id="token">──<description>──token──</description>──────────►

►─<severity>──byte──</severity>──</message>──────────────────────────────►◄
```

Tag/Fragment	Description	Type
id	unique message ID	token
description	message text	token
severity	severity of the message	byte

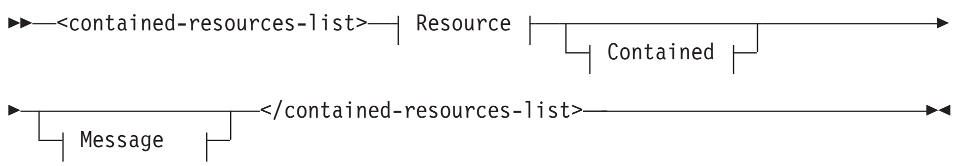
Example XML code:

```
<message id="GPM0626I">
<description>The metric 008D1000 is not defined for resource type SYSPLEX ...
</description>
<severity>4</severity>
</message>
```

Contained-Resources-List element

The <contained-resources-list> tag lists all child resources of a given resource within the resource tree.

Syntax: Contained-Resources-List



Tag/Fragment	Description	Type
Resource	description of a resource	see page 62
Contained	list of child resources	see page 64
Message	error message or warning	see page 63

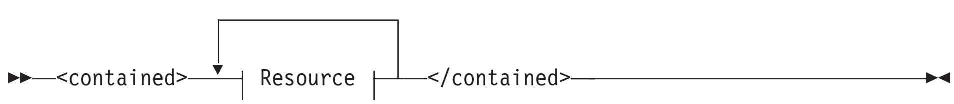
Example XML code:

```
<contained-resources-list>
<resource>
...
</resource>
<contained>
...
</contained>
</contained-resources-list>
```

Contained

The <contained> tag encloses the list of child resources.

Syntax: Contained



Tag/Fragment	Description	Type
Resource	description of a resource	see page 62

Example XML code:

```
<contained>
<resource>
<reslabel>,SYSA,MVS_IMAGE</reslabel>
<restype>MVS_IMAGE</restype>
<reslabelurl>,SYSA,MVS_IMAGE</reslabelurl>
<expandable>YES</expandable>
<icon>rmfmvsim.gif</icon>
<attributes>YES</attributes>
</resource>
...
<resource>
<reslabel>,CF01,COUPLING_FACILITY</reslabel>
<resname>CF01</resname>
<reslabelurl>,CF01,COUPLING_FACILITY</reslabelurl>
<expandable>YES</expandable>
<icon>rmfcf.gif</icon>
<attributes>YES</attributes>
</resource>
</contained>
```

Filter-Instances-List element

The <filter-instances-list> tag lists all possible filter instances with a specific list type for the given resource.

Syntax: Filter-Instances-List

```
>>--<filter-instances-list>--| Resource |------------------------------------->

                                  .------------------------------------.
                                  V                                    |
>--<listtype>--token--</listtype>---+------------------------------------+---->
                                    '--<instance>--token--</instance>--'

>--+-------------+--</filter-instances-list>---------------------------------><
   '-| Message |-'
```

Tag/Fragment	Description	Type
Resource	description of a resource	see page 62
listtype	list type of the requested filter instances	token
instance	element of the instances list	token
Message	error message or warning	see page 63

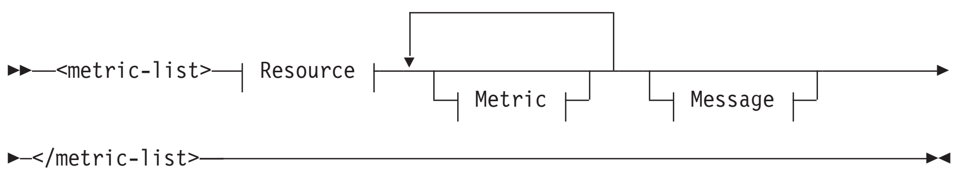
Example XML code:

```
<filter-instances-list>
<resource>
...
</resource>
<listtype>V</listtype>
<instance>SYSA,IMS610,V</instance>
...
<instance>SYSA,IMS710,V</instance>
</filter-instances-list>
```

Metric-List element

The <metric-list> tag lists all available metrics for a given resource.

Syntax: Metric-List



Tag/Fragment	Description	Type
Resource	description of a resource	see page 62
Metric	list of metrics for this resource	see page 67
Message	error message or warning	see page 63

Example XML code:

```
<metric-list>
<resource>
...
</resource>
<metric id="8D2060">
...
</metric>
...
<metric id="8D1FF0">
...
</metric>
</metric-list>
```

Metric

The <metric> tag describes a metric in detail.

Syntax: Metric

```
►►──<metric id="token">──────────────────────────────────────────────────────►

►─<description>──token──</description>──<format>──NMTOKEN──</format>────────►

►──┬─────────────────────────────────────────────┬─┬─────────────────────────────┬──►
   └─<numcols>──nonNegativeInteger──</numcols>─┘ └─<listtype>──token──</listtype>─┘

►──┬──────────────────────────────────────────────────────────────────────────┬──►
   │                         ┌──────────────────────┐                         │
   │                         ▼                      │                         │
   └─<workscopes-supported>────┤ Workscope-Type ├─────</workscopes-supported>─┘

►──┬──────────────────────────────────────┬─┬────────────────────────────┬──►
   └─<workscope>──token──</workscope>─┘ └─<filter>──token──</filter>─┘

►──┬────────────────────────┬─┬──────────────────────────────────┬──►
   └─<unit>──token──</unit>─┘ └─<helpurl>──token──</helpurl>─┘

►──┬──────────────────────────────┬─</metric>─────────────────────────────►◄
   └─<helpid>──token──</helpid>─┘
```

Workscope-Type:

```
├──<workscope-type>──NMTOKEN──</workscope-type>──────────────────────────────┤
```

Tag/Fragment	Description	Type
id	unique metric ID	token
description	description of a metric	token
format	type of counter used for this metric; value is in {single, list, report}	NMTOKEN
numcols	number of columns in report, if metric is a report	nonNegative Integer
listtype	list type of the resource	token
workscopes-supported	valid workscopes for this metric	n/a
workscope-type	valid workscope for this metric; value in {G,W,S,P,R,J}	NMTOKEN
workscope	description of the workscope (see also “The ***workscope*** parameter” on page 52)	token
filter	filter argument	token
unit	unit of the values returned by this metric; valid values in {count, index, megabytes, micro-, milliseconds , percent, rate per hour, rate per second, seconds, undefined}	token
helpurl	URL of help document	token
helpid	ID of corresponding paragraph in help document	token

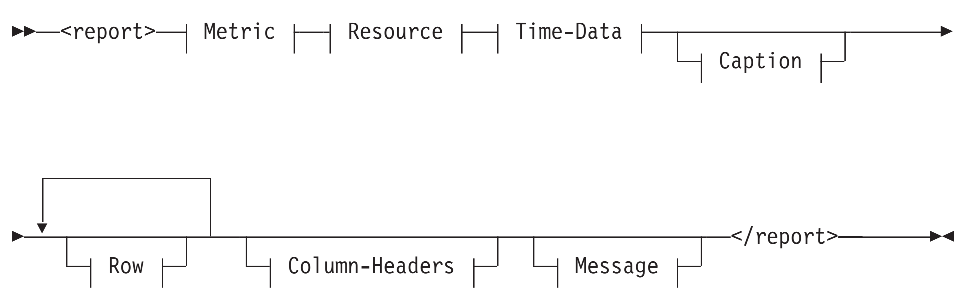
Example XML code:

```
<metric id="8D2060">
<description>% processor utilization</description>
<format>single</format>
<listtype> </listtype>
<workscopes-supported>
<workscope-type>G</workscope-type>
</workscopes-supported>
<unit>percent</unit>
<helpurl>/gpm/include/metrics.html</helpurl>
<helpid>5050</helpid>
</metric>
```

Report element

The <report> tag encloses performance data for a specific resource or metric.

Syntax: Report



Tag/Fragment	Description	Type
Metric	metric used for the report	see page 67
Resource	description of a resource	see page 62
Time-Data	time and interval information for the report	see page 69
Caption	additional (sub-) headings for the report	see page 70
Row	performance data	see page 70
Column-Headers	unique names of the columns in the report table	see page 71
Message	error message or warning	see page 63

Example XML code:

```
<report>
<metric id="CPC">
...
</metric>
<resource>
...
</resource>
<time-data>
...
</time-data>
<caption>
...
</caption>
<row refno="1">
...
</row>
```

```
...
<row refno="15">
...
</row>
<column-headers>
...
</column-headers>
</report>
```

Time-Data

The <time-data> tag provides information on the time intervals used in a report.

Syntax: Time-Data

```
►►—<time-data>—<local-start>—token—</local-start>——————————————————————►

►—<local-end>—token—</local-end>—<utc-start>—token—</utc-start>————————►

►—<utc-end>—token—</utc-end>—┬——————————————————————————————————┬———————►
                             └—<local-prev>—token—</local-prev>—┘

►—┬————————————————————————————————————┬———————————————————————————————►
  └—<local-next>—token—</local-next>—┘

►—<display-start locale="token">—token—</display-start>—————————————————►

►—<display-end locale="token">—token—</display-end">————————————————————►

►—<gatherer-interval unit="NMTOKEN">—nonNegativeInteger—</gatherer-interval>—►

►—<data-range unit="NMTOKEN">—nonNegativeInteger—</data-range>—</time-data>—►◄
```

Tag/Fragment	Description	Type
local-start	local start time of data range in format yyyymmddhhmmss (a sequence of 14 digits)	token
local-end	local end time of data range in format yyyymmddhhmmss	token
utc-start	start of data range, specified as coordinated universal time in the format yyyymmddhhmmss	token
utc-end	end of range, specified as coordinated universal time in the format yyyymmddhhmmss	token
local-prev	local timestamp of previous sample in format yyyymmddhhmmss	token
local-next	local timestamp of next sample in format yyyymmddhhmmss	token
display-start	local start time of data range in displayable format mm/dd/yyyy hh:mm:ss	token
locale	locale for which displayable format is shown	token
display-end	local end time of data range in displayable format mm/dd/yyyy hh:mm:ss	token
gatherer-interval	length of gatherer interval	nonNegative Integer
unit	unit of time; valid values in {hours, microseconds, milliseconds, minutes, seconds}	NMTOKEN
data-range	length of data range	nonNegative Integer

Example XML code:

```
<time-data>
<local-start>20110214180800</local-start>
<local-end>20110214180830</local-end>
<utc-start>20110214170800</utc-start>
<utc-end>20110214170830</utc-end>
<local-prev>20110214180745</local-prev>
<local-next>20110214180845</local-next>
<display-start locale="en-us">02/14/2011 18:08:00</display-start>
<display-end locale="en-us">02/14/2011 18:08:30</display-end>
<gatherer-interval unit="seconds">30</gatherer-interval>
<data-range unit="seconds">30</data-range>
</time-data>
```

Caption

The <caption> tag contains additional headings and summary information for a report.

Syntax: Caption

```
►►──<caption>──────────────────────────────────────────────────────────►

   ┌───────────────────────────────────────────────────────────────┐
   ▼                                                               │
►───<var>──<name>──token──</name>──<value>──token──</value>──</var>─┴───►

►─</caption>───────────────────────────────────────────────────────────►◄
```

Tag/Fragment	Description	Type
var	description of variables in additional headings	n/a
name	name of variable	token
value	value of variable	token

Example XML code:

```
<caption>
<var>
<name>CPCHPNAM</name>
<value>SYSA</value>
</var>
...
<var>
<name>CPCHMOD</name>
<value>2064</value>
</var>
</caption>
```

Row

The <row> tag contains the requested performance data values to be displayed in a Monitor III or Postprocessor report.

Syntax: Row

```
►►──<row refno="nonNegativeInteger"──┬──────────────────────┬──────────────►
                                     └─exception="token"─┘

►──┬──────────────────┬──┬─────────────────────┬──┬───────────────┬──>──────►
   └─percent="token"─┘  └─attributes="token"─┘  └─type="token"─┘

   ┌──────────────────────┐
   ▼                      │
►───<col>──token──</col>──┴──</row>──────────────────────────────────────────►◄
```

Tag/Fragment	Description	Type
col	value for cells in a row of a data table	token
refno	unique row reference number	nonNegative Integer
exception	indicates if the value in this row exceeds exception thresholds; valid values in {NONE, WARN, CRIT}	NMTOKEN
percent	percentage of the value compared to the maximum value in the list (only meaningful for list-valued metrics)	token
attributes	denotes whether attributes may be queried for the resource in this row; valid values in {YES, NO}	NMTOKEN
type	is set if the row contains values with a special meaning, for example, `SUM` denotes a row containing a total of certain previous rows	NMTOKEN

Example XML code of a row in a Postprocessor report:

```
<row refno="1" type="SUM">
<col>TOTAL/AVERAGE</col>
<col>CP</col>
<col>3.08</col>
<col>3.05</col>
<col>191.2</col>
<col>7.45</col>
<col>0.58</col>
</row>
```

Column-Headers

The <column-headers> tag contains the names of the columns in the report table:

- For Monitor III data, these names are equal to the ISPF variable names in the Monitor III ISPF reports.
- For Postprocessor data, these names are the field headings in the data tables.

Syntax: Column-Headers

```
               ┌──────────────────────────────────────────────┐
               ▼                                              │
►►──<column-headers>──<col──┬──────────────────┬──>──token──</col>─┴──────►
                            └─type="NMTOKEN"───┘

►─</column-headers>──────────────────────────────────────────────────────►◄
```

Tag/Fragment	Description	Type
col	unique name of the column	token
type	data type in this column; valid values in {N, T} for either Number or Text	NMTOKEN

Example XML code:

```
<column-headers>
<col type="T">CPCPPNAM</col>
<col type="N">CPCPDMSU</col>
...
<col type="N">CPCPLTOU</col>
</column-headers>
```

Workscope-List element

The <workscope-list> tag lists the available workscopes for a given resource.

Syntax: Workscope-List

```
►►──<workscope-list>──┤ Resource ├────────────────────────────────────────►

   ┌────────────────────────────────────┐
   ▼                                    │
►──┬────────────────────────────────────┬─┴──┬───────────────┬────────────►
   └─<workscope>──token──</workscope>───┘    └─┤ Message ├───┘

►─</workscope-list>──────────────────────────────────────────────────────►◄
```

Tag/Fragment	Description	Type
Resource	description of a resource	see page 62
workscope	description of a workscope	token
Message	error message or warning	see page 63

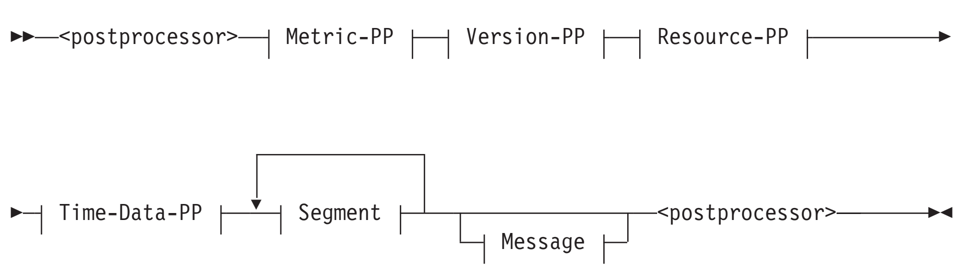
Example XML code:

```
<workscope-list>
<resource>
...
</resource>
<workscope>,BATCH,W</workscope>
...
<workscope>,HOTTSO,S</workscope>
</workscope-list>
```

Postprocessor element

The <postprocessor> tag encloses one Postprocessor report denoted by the Metric-PP specification.

Syntax: Postprocessor



Tag/Fragment	Description	Type
Metric-PP	name of a Postprocessor report	see page 73
Version-PP	version information for Postprocessor reports	see page 74
Resource-PP	description of the reported resource	see page 75
Time-Data-PP	time information for the report interval	see page 75
Segment	named Postprocessor report segment containing the report data	see page 76
Message	error message or warning	see page 63

Example XML code:

```
<postprocessor>
<metric id="CPU">...</metric>
<version>...</version>
<resource>...</resource>
<time-data>...</time-data>
&lt;segment&gt;<name>CPU ACTIVITY</name>...&lt;/segment&gt;
&lt;segment&gt;<name>PARTITION DATA REPORT</name>...&lt;/segment&gt;
&lt;segment&gt;<name>LPAR CLUSTER REPORT</name>...&lt;/segment&gt;
&lt;segment&gt;<name>GROUP CAPACITY REPORT</name>...&lt;/segment&gt;
</postprocessor>
```

Metric-PP

The <metric> tag within the <postprocessor> tag contains the name of a Postprocessor report.

Syntax: Metric-PP

```
►►──<metric──┬──────────────┬──┬──────────────────┬──>──────────────────────►
             └─id="token"───┘  └─option="token"───┘

►─<description>──token──</description>──<type>──token──</type>──────────────►

►──┬───────────────────────────────┬──┬─────────────────────────────┬───────►
   └─<helpurl>──token──</helpurl>──┘  └─<helpid>──token──</helpid>──┘

►─</metric>─────────────────────────────────────────────────────────────────►◄
```

Tag/Fragment	Description	Type
id	name of the Postprocessor report	token
option	suboptions specified for the Postprocessor reports	token
description	descriptive title of the report	token
type	type of the Postprocessor report; can be either Interval, Duration, or Overview	token
helpurl	URL of help document	token
helpid	ID of corresponding paragraph in help document	token

Example XML code:

```
<metric id="CPU">
<description>CPU Activity Report</description>
<type>Interval</type>
</metric>
```

Version-PP

The <version> tag within the <postprocessor> tag contains version information for Postprocessor reports.

Syntax: Version

```
►►──<version>──<smf_data>──token──</smf_data>──<rmf_report>──token──</rmf_report>──</version>──────►◄
```

Tag/Fragment	Description	Type
smf_data	version of the operating system that captured the SMF data	token
rmf_report	version of the RMF Postprocessor	token

Example XML code:

```
<version>
<smf-data>z/OS V1R12</smf-data>
<rmf-report>V1R12 RMF</rmf-report>
</version>
```

Resource-PP

A <resource> tag within the <postprocessor> tag specifies the reported resource, for example, a system.

Syntax: Resource-PP

```
►►──<resource>──<resname>──token──</resname>──────────────────────────►

►─<restype>──token──</restype>─┬──────────────────────────────┬───────►
                               └─<helpurl>──token──</helpurl>─┘

►─</resource>─────────────────────────────────────────────────────────►◄
```

Tag/Fragment	Description	Type
resname	resource name	token
restype	resource type	token

Example XML code:

```
<resource>
<resname>SYSE</resname>
<restype>SYSTEM ID</restype>
</resource>
```

Time-Data-PP

A <time-data> tag within the <postprocessor> tag provides information on the time intervals used in a report.

Syntax: Time-Data-PP

```
►►──<time-data>──<display-start locale="token">──token──</display-start>──────►

►─<display-end locale="token">──token──</display-end">────────────────────────►

►─<report-interval unit="NMTOKEN">──nonNegativeInteger──</report-interval>────►

►─┬───────────────────────────────────────────────────────┬───────────────────►◄
  └─<cycle unit="NMTOKEN">──nonNegativeInteger──</cycle>─┘
```

Tag/Fragment	Description	Type
display-start	local start date and time of the reporting interval in displayable format `mm/dd/yyyy-hh:mm:ss`	token
locale	locale for which displayable format is shown	token
display-end	local end date and time of the reporting interval in displayable format `mm/dd/yyyy-hh:mm:ss`	token

Tag/Fragment	Description	Type
report-interval	length of the measurement interval (reporting time)	nonNegative Integer
cycle	cycle length of data sampling	nonNegative Integer
unit	unit of time; valid values in {hours, minutes, seconds, milliseconds, microseconds}	NMTOKEN

Example XML code:

```
<time-data>
<display-start locale="en-us">03/26/2010-11.15.00</display-start>
<display-end locale="en-us">03/26/2010-11.30.00</display-end>
<report-interval unit="minutes">14:59:999</report-interval>
<cycle unit="milliseconds">1000</cycle>
</time-data>
```

Segment

A <segment> tag contains one report section of a Postprocessor report.

Syntax: Segment

```
                                          ┌──────────┐
                                          ▼          │
►►──&lt;segment&gt;──┬────────────────────────┬──┤ Part ├──┴──┬───────────────┬──►
               └─<name>──token──</name>─┘               └──┤ Message ├──┘

►──&lt;/segment&gt;──────────────────────────────────────────────────────────────►◄
```

Tag/Fragment	Description	Type
name	name of a report segment	token
Part	part of a report segment which can contain either a variable-value list or a data table	see page 76
Message	error message or warning	see page 63

Example XML code:

```
&lt;segment&gt;<name>CPU ACTIVITY</name>
<part>...</part>
<part>...</part>
...
&lt;/segment&gt;
```

Part

A <part> tag encloses a part of a report segment. Such a part can either be a variable-value list (specified by the ***Var*** fragment) or a data table (specified by the ***Column-Headers*** and ***Row*** fragments).

Syntax: Part

```
►►──<part>──┬──────────────────────────────┬──────────────────────────►
            └─<name>──token──</name>─┘

     ┌──────────┐
     ▼          │
►──┬──┤ Var ├───┴──────────────────────┬─</part>────────────────────►◄
   │                       ┌────────┐  │
   │                       ▼        │  │
   └─┤ Column-Headers ├──┤ Row ├────┴──┘
```

Tag/Fragment	Description	Type
name	name of a part within a report segment	token
Var	a variable-value pair	see page 77
Column-Headers	unique names of the columns in the report table	see page 71
Row	performance data	see page 70

Example XML code:

see the ***Example XML code*** for the <var> tag.

Var

A <var> tag contains a variable-value pair, for example, `CPU 2097`.

Syntax: Var

```
►►──<var>──<name>──token──</name>──<value>──token──</value>──</var>──────►◄
```

Tag/Fragment	Description	Type
name	name of a variable	token
value	value of a variable	token

Example XML code:

```
<part>
<var><name>CPU</name><value>2097</value></var>
<var><name>Model</name><value>729</value></var>
...
</part>
<part>
<column-headers>
<col>CPU Number</col>
<col>CPU Type</col>
<col>Time% Online</col>
<col>Time% LPAR Busy</col>
<col>Time% MVS Busy</col>
</column-headers>
```

```
<row refno="1">
<col>0</col>
<col>CP</col>
<col>100.00</col>
<col>3.12</col>
<col>3.08</col>
</row>
<row refno="2">...</row>
...
</part>
...
<part><name>System Address Space Analysis</name>
<var><name>Samples</name><value>900</value></var>
</part>
```

Coding example for requesting and receiving Monitor III performance data

The subsequent Java code sample demonstrates a method how to send a HTTP requests to the DDS and how to receive the response into a text file, from where you can extract the required values.

```
import java.io.*;
import java.net.*;
import sun.misc.*;

/*************************************************************
 * A command line program to communicate with the DDS server  *
 *************************************************************/

public class ParseDDS {

public final static String URLPREFIX =
       "http://<ddshost>:<ddsport>/gpm/";
public final static String OUTFILE = "ParseDDS.txt";

PrintWriter writer;

/***************************************************************************
 * Constructor: Builds a performance data request and sends it to the DDS. *
 * Writes results into a local file.                                       *
 ***************************************************************************/

public ParseDDS() {
 try {
       String urlstr;
       writer = new PrintWriter(new FileOutputStream(OUTFILE, false));
       for (int i = 0; i < 1; i++) {

       /* Get performance data (Performance index of all service class periods) */

       urlstr = URLPREFIX + "perform/perform.xml?resource=,*,SYSPLEX&id=8D1020";
       getAndWrite(urlstr);

       }
 }
 catch (Exception e) {
        e.printStackTrace();
 }

 writer.close();

}

/*******************************************************
 * Sends request to server and writes XML data to file  *
 *******************************************************/
```

```
public void getAndWrite(String urlstr) throws Exception {
       URL url = new URL(urlstr);
       HttpURLConnection con = (HttpURLConnection) url.openConnection();
       String line;
       con.setDoInput(true);
       con.connect();
       int cl = con.getContentLength();

       /* Write HTTP contents (XML document) to file */

       line = "URL=" + urlstr;
       System.out.println(line);
       writer.println(line);
       if (cl > 0) {
              BufferedReader reader =
                 new BufferedReader(new InputStreamReader(con.getInputStream()));
              int k = 0;
              while ((line = reader.readLine()) != null) {
                      k++;
                      System.out.println(line);
                      writer.println(line);
              }
              System.out.println(
                      "   " + cl + " Bytes XML content received from DDS.");
       }
       System.out.println(" ");
}

/************************************************************
 *                            main                          *
 ************************************************************/

public static void main(String[] args) {

     System.out.println(
             "ParseDDS started. Output will be written to file: " + OUTFILE);
     ParseDDS myDDs = new ParseDDS();
     System.out.println("ParseDDS ended.");
     System.exit(0);
  }
}
```

Chapter 4. z/OS CIM monitoring

The Common Information Model (CIM) is a standard data model developed by a consortium of major hardware and software vendors (including IBM), called the Distributed Management Task Force (DMTF). It provides a common definition for describing and accessing systems management data in heterogeneous environments. It allows vendors and system programmers to write applications (CIM monitoring clients) that measure system resources in a network with different operating systems and hardware, and to actually manage those systems.

The z/OS base element ***Common Information Model*** (z/OS CIM) implements the ***CIM server***, based on the OpenPegasus open source project. A CIM monitoring client invokes the CIM server who in turn collects z/OS metrics from the system and returns it to the calling client.

A part of z/OS CIM is the ***eServer OS management profile***. It provides access to, and management of, the base IT resources, like zSeries LPARs, z/OS operating system images, and z/OS address spaces.

A part of z/OS RMF is the ***eServer OS monitoring profile***. It provides monitoring data for the IT resources exposed by the ***eServer OS management profile***. This profile allows read access to monitoring data that pertains to the current time, and also provides navigation between the resources and their monitoring data.

If a CIM client requests the CIM server to obtain z/OS metrics, the CIM server invokes the appropriate ***z/OS RMF monitoring provider*** which retrieves these metrics associated to z/OS system resources. The ***z/OS RMF monitoring providers*** use RMF Monitor III performance data.

For more information on the z/OS CIM element refer to *z/OS Common Information Model User's Guide* (SC33-7998).

A Java sample program demonstrating the use of CIM indications is provided in "Sample program to subscribe for and to receive indications" on page 118.

eServer OS monitoring

Note:
eServer OS monitoring for z/OS is only available if RMF is running and the RMF DDS is started. How to provide these prerequisites is described in *Starting and stopping RMF*, in *Setting up the Distributed Data Server*, and in *Starting the Distributed Data Server* in the *z/OS RMF User's Guide*. How to set the required environment variables for the CIM server is described in the *z/OS Common Information Model User's Guide* (SC33-7998).

eServer OS monitoring uses metrics that are associated with resource classes implemented by eServer OS management providers described in the *z/OS Common Information Model User's Guide*.

The following CIM classes have been implemented as IBM supplied providers according to the DMTF dynamic metrics model. You can find more information on this data model in the CIM Metrics White Paper (DSP0141), which is available at the DMTF web page (http://www.dmtf.org) in the CIM White Paper section.

- ***IBMzOS_BaseMetricDefinition***: representation of metric definition (for example, metric name and ID); a subclass of ***CIM_BaseMetricDefinition***
- ***IBMzOS_BaseMetricValue***: representation of a metric value (with value, time stamp and duration); a subclass of ***CIM_BaseMetricValue***
- ***IBMzOS_MetricDefForME***: association between a managed element (resource) and metric definition instances; a subclass of ***CIM_MetricDefForME***.
- ***IBMzOS_MetricForME***: association between a managed element (resource) and metric value instances; a subclass of ***CIM_MetricForME***.
- IBMzOS_BaseMetricInstance: association between metric definition and metric value instances; a subclass of ***CIM_MetricInstance***.

All instances of IBMzOS_BaseMetricValue will return volatile data. Historical data is not supported with this release.

For a list of the metrics supported in z/OS, see "z/OS metrics" on page 87. Also see the CIM class and instance MOF files. The MOF file *IBMzOS_Monitoring.mof* can be found in `/usr/lpp/wbem/provider/schemas/os_monitoring`.

The following diagram illustrates the relationship between the IBM extension classes, and the CIM base classes they extend:

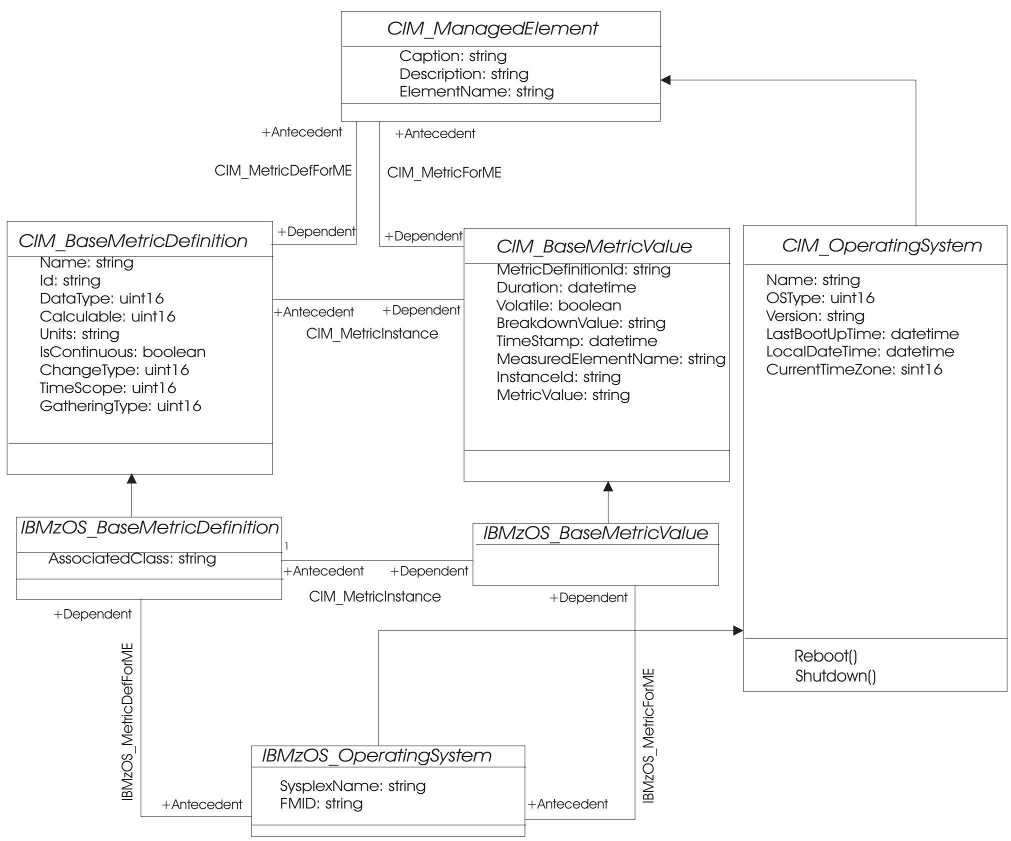


Figure 13. z/OS RMF implementation of the DMTF dynamic metrics model

Class ***IBMzOS_OperatingSystem*** is implemented by the *z/OS OS Management Providers* (part of base element z/OS CIM). It is just an example of a resource which can be inherited from ***CIM_OperatingSystem***. Class ***CIM_OperatingSystem*** can be implemented with the same attributes on any operating system that exploits the CIM model. ***IBMzOS_OperatingSystem*** inherits all properties from this base class and implements further z/OS-specific attributes, like `SysplexName` and `FMID`. BaseMetricValue and BaseMetricDefinition instances can be associated to it.

CIM_BaseMetricDefinition:

An instance of this class represents how a metric is defined. The associated class CIM_BaseMetricValue holds the metric value. CIM_BaseMetricDefinition provides a way to introduce a new metric definition at run time and capture its instance values in a separate class.

CIM_BaseMetricValue:

Each instance of this class represents a metric value.

CIM_MetricDefForME/IBMzOS_MetricDefForME:

Traversing this association from a resource returns the set of all metric definitions for the given resource. **Usage example:** If you want to know what metric definitions are available for the ***CIM_Process*** resource, you can use this association. This association returns static data, which does not change without applying service to z/OS CIM. Therefore call it once in your application at startup time to figure out what is available. You do not need to traverse it several times.

CIM_MetricForME/IBMzOS_MetricForME:

This association links a measured element (resource instance) to all metric instances available for it. **Usage example:** Traversing this association starting from an ***IBMzOS_OperatingSystem*** instance, returns all associated ***IBMzOS_BaseMetricValue*** instances. In other words, this association returns metrics for the z/OS image on which the CIM server is running.

CIM_MetricInstance/IBMzOS_MetricInstance:

Traversing this association gives you all metric instances available for a given ***CIM_BaseMetricDefinition***. **Usage example:** In order to get the metric values for the ResidentSetSize (working set size) of all process instances (z/OS address spaces) you can start at the metric definition instance of the ResidentSetSize instance. Traversing the ***CIM_MetricInstance*** association returns all instances of the ResidentSetSize metric for all address spaces in your system.

The above classes and associations are described in more detail and with all attributes in the ***eServer CIM*** document:

`http://publib.boulder.ibm.com/infocenter/eserver/v1r1/en_US/info/ciminfo/eicah.pdf`

z/OS resource classes based on RMF

This section documents those CIM resource classes available for z/OS that are implemented based on RMF data.

Note: All described classes are only available if RMF is up and running on the system where the monitored resources are located.

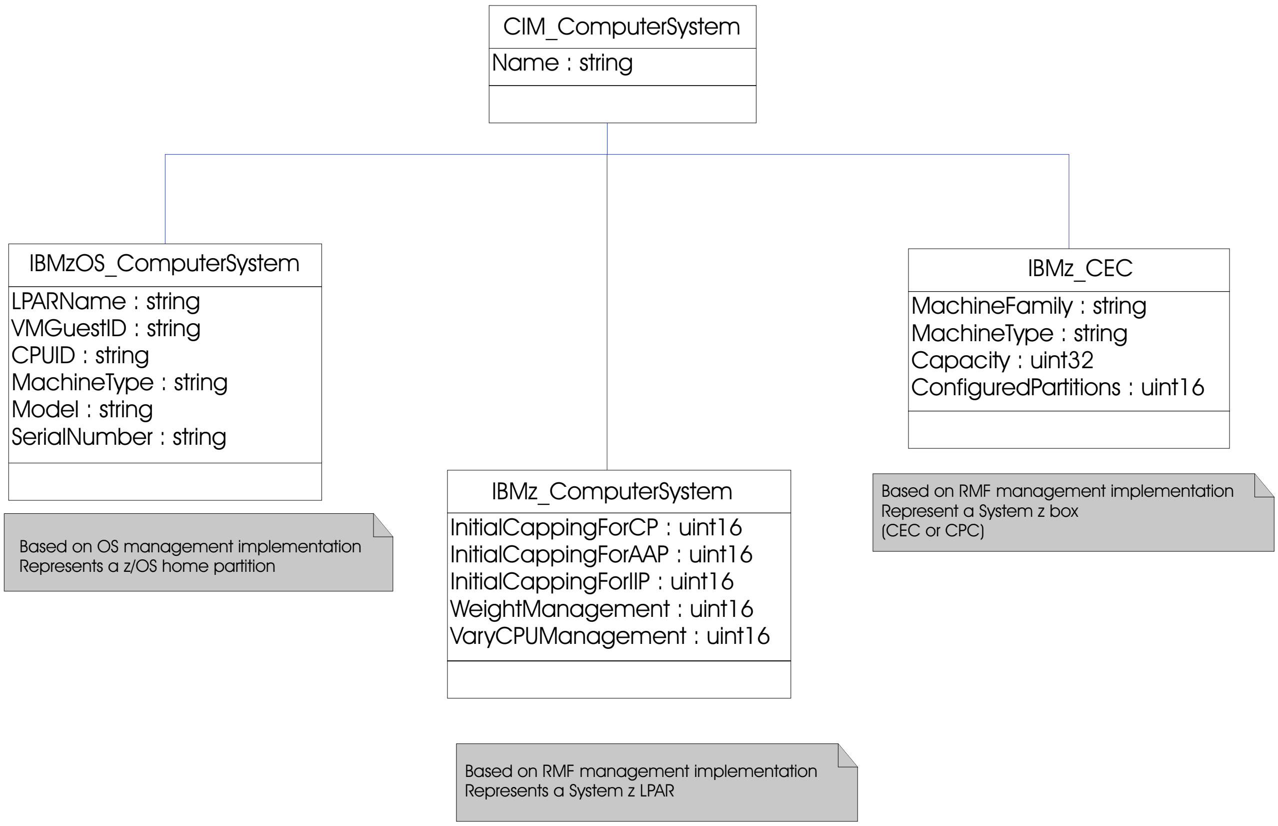


Figure 14. z/OS resource classes implemented by RMF. Simplified subset of LPAR- and CEC-related resource classes

IBMz_CEC

This resource represents a System z box including processors, memory, I/O cages and so on. Note that CEC (central electronics complex) is a commonly used synonym for CPC (central processing complex). IBMz_CEC contains the following attributes:

Property	Description
string Name (key)	CEC serial number
string CreationClassName (key)	Value “IBMz_CEC”
string Machine Family	Processor type, for example, “2094”
string Machine Type	Software model of the processor, for example, “716”
uint32 Capacity	System capacity in MSU/hour
uint16 ConfiguredPartitions	Number of configured partitions in the CEC

IBMz_ComputerSystem

This class represents a logical partition (LPAR) and contains the following attributes:

Property	Description
string Name (key)	LPAR name
string CreationClassName (key)	Value “IBMz_ComputerSystem”
string CECName (key)	Name of the CEC this LPAR exists on
uint16 InitialCappingForCP	Information about initial capping for this partition (logical processor type CP): 0: unknown, 1: other, 2: enabled, 3: disabled
uint16 InitialCappingForAAP	Information about initial capping for this partition (logical processor type zAAP): 0: unknown, 1: other, 2: enabled, 3: disabled
uint16 InitialCappingForIIP	Information about initial capping for this partition (logical processor type zIIP): 0: unknown, 1: other, 2: enabled, 3: disabled
uint16 WeightManagement	Information about z/OS IRD LPAR weight management: 0: unknown, 1: other, 2: enabled, 3: disabled
uint16 VaryCPUManagement	Information about z/OS IRD vary CPU management: 0: unknown, 1: other, 2: enabled, 3: disabled
uint16 zAAPHonorPriority	Information about zAAP honor priority: 0: unknown, 1: other, 2: enabled, 3: disabled
uint16 zIIPHonorPriority	Information about zIIP honor priority: 0: unknown, 1: other, 2: enabled, 3: disabled

IBMzOS_Channel

This resource represents a channel path in the computer system, based on RMF information. IBMzOS_Channel contains the following attributes:

Property	Description
string DeviceID (key)	Channel path ID (CHPID)
string SystemName (key)	z/OS MVS system name
string ChannelPathType	Type of channel path

IBMzOS_WLMServiceDefinition

This class represents the z/OS WLM policy. It is a subclass from ***CIM_ManagedElement*** and contains the following attributes:

Property	Description
string Name (key)	Name of the WLM service definition
string PolicyName	Name of the active WLM service policy
datetime PolicyActivationTime	Date and time the WLM service policy has been activated

IBMzOS_WLMServiceClassPeriod

This class provides basic properties of a service class period defined for a WLM service class and contains the following attributes:

Property	Description
string ServiceClassName	Name of the WLM service class to which this service class period belongs
string PeriodNumber	Service class period in which the service class is currently running
uint16 ImportanceLevel	Importance level ranging form 1 to 5 where 1 is the most important level. If the property value cannot be determined, for example, for inactive service classes periods or for service class periods with importance 'discretionary', NULL is returned.

z/OS metrics

This chapter describes the CIM metrics, as they are supported by z/OS RMF, with z/OS specific details. Most of the metrics are eServer metrics. If a metric is z/OS specific, this is explicitly mentioned here. Some important details regarding the implementation on other platforms are mentioned as well, so you can use this overview to start writing cross-platform monitoring applications.

In the following, this chapter contains a subsection for each resource class which lists the metrics available for these classes.

IBMzOS_LogicalDisk

This class represents a logical disk in the system. The following metrics are associated to IBMzOS_LogicalDisk:

ActiveTimePercentage			
Description: Percentage of time the disk unit was actively processing requests, calculated as the sum of connect, disconnect and pending time. This metric cannot exceed 100%, and it does not state how many requests were active at any given time.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% active time	***DDS metric ID***	8D0010
Associated class	IBMzOS_LogicalDisk		
AvailableSpace			
Description: Free capacity on the disk drive.			
Datatype	uint64	***Units***	bytes
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	freespace	***DDS metric ID***	8D3090
Associated class	IBMzOS_LogicalDisk		
AverageDeviceUtilization			
Description: Average device utilization (not normalized to 100% for parallel I/O activity). This metric can be above 100% if the device is executing multiple I/Os in parallel. A busy time percentage of 200% means that on average, the device was executing two I/Os in parallel. This metric does not tell you whether the busy time of the measured device could be even higher, because this is architecture dependent.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	i/o activity rate and response time	***DDS metric ID***	based on 8D0E90 and 8D10F0

Associated class	IBMzOS_LogicalDisk		
Capacity			
Description: Capacity of the disk drive.			
Datatype	uint64	***Units***	bytes
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	capacity	***DDS metric ID***	8D2FF0
Associated class	IBMzOS_LogicalDisk		
FastWriteOperations			
Description: Number of write operations executed as fast writes. With fast writes, the data is first stored in non volatile memory instead of being stored directly on disk, which leads to better performance. This metric is an eServer metric and currently not implemented for z/OS.			
Datatype	uint64	***Units***	requests
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	not implemented	***DDS metric ID***	not implemented
Associated class	IBMzOS_LogicalDisk		
FastWritePercentage			
Description: Percentage of write operations executed as fast writes.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% cache DFW hits (all systems)	***DDS metric ID***	8D21D0
Associated class	IBMzOS_LogicalDisk		
IOIntensity			
Description: I/O utilization indicator: IOIntensity = ResponseTime * IORate			
Datatype	real32	***Units***	milliseconds per second
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	i/o intensity	***DDS metric ID***	8D1290
Associated class	IBMzOS_LogicalDisk		
QueueDepth			
Description: Average number of I/O requests currently in queue (OS view).			

Datatype	real32	*Units*	s/s = I/O request rate [1/s] * average response time [ms] / 1000
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	i/o activity rate and response time	*DDS metric ID*	based on 8D0E90 and 8D10F0
Associated class	IBMzOS_LogicalDisk		
ReadCacheHitPercentage			
Description: Percentage of read requests that did not need access to disk drives because data was available in cache.			
Datatype	real32	*Units*	percent
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	% cache read hits (all systems)	*DDS metric ID*	8D2280
Associated class	IBMzOS_LogicalDisk		
ReadOperations			
Description: Number of read operations issued from the system against the disk drive within the interval. One request can transfer more than one block. The read operation may or may not require physical disk operation. This metric is an eServer metric and currently not implemented for z/OS.			
Datatype	uint64	*Units*	requests
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	not implemented	*DDS metric ID*	not implemented
Associated class	IBMzOS_LogicalDisk		
ReadThroughput			
Description: Bytes read per second. This metric is an eServer metric and currently not implemented for z/OS.			
Datatype	uint64	*Units*	bytes per second
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	not implemented	*DDS metric ID*	not implemented
Associated class	IBMzOS_LogicalDisk		
RequestRate			
Description: Number of I/O requests per second for the associated device.			
Datatype	real32	*Units*	1 / s (events per second)

ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	i/o activity rate	*DDS metric ID*	8D0E90
Associated class	IBMzOS_LogicalDisk		
ResponseTime			
Description: ResponseTime associated to a logical disk drive. The average response time (in milliseconds) that the device required to complete an I/O request. For comparison of average response times on different platforms, please keep in mind that this metric may be reported by the disk device itself or it may be computed by the operating system, beginning at the point in time when an application issues a disk related command and ending at the point in time when the data is returned. In this case, the queue times, network times and other components of response time are included in the ResponseTime metric. In z/OS, this metric represents the operating system view of the disk response time.			
Datatype	real32	*Units*	milliseconds
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	response time	*DDS metric ID*	8D10F0
Associated class	IBMzOS_LogicalDisk		
TransferredThroughput			
Description: Bytes transferred per second. This metric is an eServer metric and currently not implemented for z/OS.			
Datatype	uint64	*Units*	bytes per second
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	not implemented	*DDS metric ID*	not implemented
Associated class	IBMzOS_LogicalDisk		
WaitTime			
Description: WaitTime associated to a logical disk drive. This metric comprises an estimation of the delay components of ***ResponseTime*** (in milliseconds), or ***AverageResponseTime*** minus service time, and the time spent in queues.			
Datatype	real32	*Units*	milliseconds
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	% connect time and response time	*DDS metric ID*	based on 8D00B0 and 8D10F0
Associated class	IBMzOS_LogicalDisk		
WriteOperations			

Description: Total number of write operations executed within the interval. One request can transfer more than one block. This metric is an eServer metric and currently not implemented for z/OS.			
Datatype	uint64	***Units***	requests
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	not implemented	***DDS metric ID***	not implemented
Associated class	IBMzOS_LogicalDisk		
WriteThroughput			
Description: Bytes written per second. This information is not available on logical disk granularity, only on physical RAID rank level. This metric is an eServer metric and currently not implemented for z/OS.			
Datatype	uint64	***Units***	bytes per second
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	not implemented	***DDS metric ID***	not implemented
Associated class	IBMzOS_LogicalDisk		

IBMz_CEC

This class contains basic properties of a CEC box of a System z.

Note: Not only that CEC is instrumented on which RMF is running, but all CECs of the sysplex. If z/OS is running as a guest under z/VM, **IBMz_CEC** instances and associated metrics are not available.

LPARWeightForAAP			
Description: LPAR weight for processor type zAAP.			
Datatype	uint32	***Units***	weight
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	LPAR weight (AAP)	***DDS metric ID***	8D3F30
Associated class	IBMz_CEC		
LPARWeightForCP			
Description: LPAR weight for standard processor.			
Datatype	uint32	***Units***	weight
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true

DDS metric name	LPAR weight (CP)	***DDS metric ID***	8D3F60
Associated class	IBMz_CEC		
LPARWeightForICF			
Description: LPAR weight for processor type ICF.			
Datatype	uint32	***Units***	weight
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	LPAR weight (ICF)	***DDS metric ID***	8D3F90
Associated class	IBMz_CEC		
LPARWeightForIFL			
Description: LPAR weight for processor type IFL.			
Datatype	uint32	***Units***	weight
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	LPAR weight (IFL)	***DDS metric ID***	8D3FE0
Associated class	IBMz_CEC		
LPARWeightForIIP			
Description: LPAR weight for processor type zIIP.			
Datatype	uint32	***Units***	weight
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	LPAR weight (IIP)	***DDS metric ID***	8D4010
Associated class	IBMz_CEC		
NumberOfDedicatedCPs			
Description: Number of dedicated standard processors.			
Datatype	uint32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	none	***DDS metric ID***	none
Associated class	IBMz_CEC		
NumberOfDefinedAAPs			
Description: Number of defined processors of type zAAP.			
Datatype	uint32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true

DDS metric name	none	***DDS metric ID***	none
Associated class	IBMz_CEC		
NumberOfDefinedCPs			
Description: Number of defined standard processors.			
Datatype	uint32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	none	***DDS metric ID***	none
Associated class	IBMz_CEC		
NumberOfDefinedICFs			
Description: Number of defined processors of type ICF.			
Datatype	uint32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	none	***DDS metric ID***	none
Associated class	IBMz_CEC		
NumberOfDefinedIFLs			
Description: Number of defined processors of type IFL.			
Datatype	uint32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	none	***DDS metric ID***	none
Associated class	IBMz_CEC		
NumberOfDefinedIIPs			
Description: Number of defined processors of type zIIP.			
Datatype	uint32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	none	***DDS metric ID***	none
Associated class	IBMz_CEC		
NumberOfSharedAAPs			
Description: Number of shared processors of type zAAP.			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true

DDS metric name	none	*DDS metric ID*	none
Associated class	IBMz_CEC		
NumberOfSharedCPs			
Description: Number of shared standard processors.			
Datatype	uint32	*Units*	processors
ChangeType	4 (Gauge)	*TimeScope*	2 (Point)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	none	*DDS metric ID*	none
Associated class	IBMz_CEC		
NumberOfSharedIIPs			
Description: Number of zIIPs in zIIP shared pool (shared physicals).			
Datatype	uint32	*Units*	processors
ChangeType	4 (Gauge)	*TimeScope*	2 (Point)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	none	*DDS metric ID*	none
Associated class	IBMz_CEC		
SumOfAAPsAcrossLPARs			
Description: Sum of zAAPs across all LPARs in CEC (shared logicals).			
Datatype	uint32	*Units*	processors
ChangeType	4 (Gauge)	*TimeScope*	2 (Point)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	# logical processors shared (AAP)	*DDS metric ID*	8D3B70
Associated class	IBMz_CEC		
SumOfCPsAcrossLPARs			
Description: Sum of CPs across all LPARs in CEC (shared logicals).			
Datatype	uint32	*Units*	processors
ChangeType	4 (Gauge)	*TimeScope*	2 (Point)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	# logical processors shared (CP)	*DDS metric ID*	8D3BA0
Associated class	IBMz_CEC		
SumOfIIPsAcrossLPARs			
Description: Sum of zIIPs across all LPARs in CEC (shared logicals).			
Datatype	uint32	*Units*	processors
ChangeType	4 (Gauge)	*TimeScope*	2 (Point)
GatheringType	3 (Periodic)	*IsContinuous*	true

DDS metric name	# logical processors shared (IIP)	***DDS metric ID***	8D3C50
Associated class	IBMz_CEC		
SumOfOnlineAAPsAcrossLPARs			
Description: Sum of online zAAPs across all LPARs in CEC (shared logicals).			
Datatype	uint32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors online (AAP)	***DDS metric ID***	8D3C80
Associated class	IBMz_CEC		
SumOfOnlineCPsAcrossLPARs			
Description: Sum of online CPs across all LPARs in CEC (shared logicals).			
Datatype	uint32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors online (CP)	***DDS metric ID***	8D3CA0
Associated class	IBMz_CEC		
SumOfOnlineIIPsAcrossLPARs			
Description: Sum of online zIIPs across all LPARs in CEC (shared logicals).			
Datatype	uint32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors online (IIP)	***DDS metric ID***	8D3D50
Associated class	IBMz_CEC		
TotalAAPTimePercentage			
Description: Total physical zAAP utilization percentage (CEC level).			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of total physical utilization (AAP)	***DDS metric ID***	8D3300
Associated class	IBMz_CEC		
TotalCPTimePercentage			
Description: Total physical CP utilization percentage (CEC level).			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true

DDS metric name	% of total physical utilization (CP)	***DDS metric ID***	8D2540
Associated class	IBMz_CEC		
TotalIIPTimePercentage			
Description: Total physical zIIP utilization percentage (CEC level).			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of total physical utilization (IIP)	***DDS metric ID***	8D1C70
Associated class	IBMz_CEC		
TotalSharedAAPTimePercentage			
Description: Total physical IBM System z9 Application Assist Processor utilization percentage (shared zAAP, CEC level).			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of total physical utilization (AAP)	***DDS metric ID***	8D3910
Associated class	IBMz_CEC		
TotalSharedCPTimePercentage			
Description: Total physical CP utilization percentage (shared CP, CEC level).			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of total physical utilization (shared CP)	***DDS metric ID***	8D3920
Associated class	IBMz_CEC		
TotalSharedICFTimePercentage			
Description: Total physical ICF utilization percentage (shared ICF, CEC level).			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of total physical utilization (shared ICF)	***DDS metric ID***	8D3930
Associated class	IBMz_CEC		
TotalSharedIFLTimePercentage			
Description: Total physical IFL utilization percentage (shared IFL, CEC level).			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)

GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of total physical utilization (shared IFL)	***DDS metric ID***	8D3940
Associated class	IBMz_CEC		
TotalSharedIIPTimePercentage			
Description: Total physical zIIP utilization percentage (shared zIIP, CEC level).			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of total physical utilization (shared IIP)	***DDS metric ID***	8D3950
Associated class	IBMz_CEC		

IBMz_ComputerSystem

Instances of the ***IBMz_ComputerSystem*** class represent System z logical partitions (LPARs). If z/OS is running under z/VM, ***IBMz_ComputerSystem*** instances and associated metrics are not available.

LPARWeightForAAP			
Description: LPAR weight (processor type zAAP).			
Datatype	uint32	***Units***	weight
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	LPAR weight (AAP)	***DDS metric ID***	8D3F20
Associated class	IBMz_ComputerSystem		
LPARWeightForCP			
Description: LPAR weight (processor type CP).			
Datatype	uint32	***Units***	weight
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	LPAR weight (CP)	***DDS metric ID***	8D3F50
Associated class	IBMz_ComputerSystem		
LPARWeightForICF			
Description: LPAR weight (processor type ICF).			
Datatype	uint32	***Units***	weight
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	LPAR weight (ICF)	***DDS metric ID***	8D3F80

Associated class	IBMz_ComputerSystem		
LPARWeightForIIP			
Description: LPAR weight (processor type zIIP).			
Datatype	uint32	***Units***	weight
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	LPAR weight (IIP)	***DDS metric ID***	8D4000
Associated class	IBMz_ComputerSystem		
NumberOfDedicatedAAPs			
Description: Number of dedicated processors of type zAAP.			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors dedicated (AAP)	***DDS metric ID***	8D3B00
Associated class	IBMz_ComputerSystem		
NumberOfDedicatedCPs			
Description: Number of dedicated standard processors.			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors dedicated (CP)	***DDS metric ID***	8D3B20
Associated class	IBMz_ComputerSystem		
NumberOfDedicatedIIPs			
Description: Number of dedicated processors of type zIIP.			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors dedicated (IIP)	***DDS metric ID***	8D3B40
Associated class	IBMz_ComputerSystem		
NumberOfOnlineAAPs			
Description: Number of online processors of type zAAP.			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors online (AAP)	***DDS metric ID***	8D3C70

Associated class	IBMz_ComputerSystem		
NumberOfOnlineCPs			
Description: Number of online standard processors.			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors online (CP)	***DDS metric ID***	8D2610
Associated class	IBMz_ComputerSystem		
NumberOfOnlineICFs			
Description: Number of online processors of type ICF.			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors online (ICF)	***DDS metric ID***	8D3CC0
Associated class	IBMz_ComputerSystem		
NumberOfOnlineIFLs			
Description: Number of online processors of type IFL.			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors online (IFL)	***DDS metric ID***	8D3D10
Associated class	IBMz_ComputerSystem		
NumberOfOnlineIIPs			
Description: Number of online processors of type zIIP.			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# processors online (IIP)	***DDS metric ID***	8D3D40
Associated class	IBMz_ComputerSystem		
NumberOfSharedAAPs			
Description: Number of shared processors of type zAAP (shared logicals).			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# logical processors shared (AAP)	***DDS metric ID***	8D3B60

Associated class	IBMz_ComputerSystem		
NumberOfSharedCPs			
Description: Number of shared standard processors (shared logicals).			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# logical processors shared (CP)	***DDS metric ID***	8D3B90
Associated class	IBMz_ComputerSystem		
NumberOfSharedIIPs			
Description: Number of shared processors of type zIIP (shared logicals).			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# logical processors shared (IIP)	***DDS metric ID***	8D3C40
Associated class	IBMz_ComputerSystem		
PartitionCapacityCappedPercentage			
Description: WLM capping percentage.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% WLM capping	***DDS metric ID***	8D2490
Associated class	IBMz_ComputerSystem		
PartitionCapacityFourHourAverage			
Description: Four hour rolling average.			
Datatype	uint64	***Units***	MSU/h
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	four hour MSU average	***DDS metric ID***	8D2630
Associated class	IBMz_ComputerSystem		
PartitionDefinedCapacity			
Description: Percentage of defined CPU capacity used by this LPAR or VM guest.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	defined MSU	***DDS metric ID***	8D2620

Associated class	IBMzOS_ComputerSystem		
PartitionDefinedCapacityUsedPercentage			
Description: Percentage of defined CPU capacity used by this LPAR or z/VM guest.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% capacity used	***DDS metric ID***	8D2870
Associated class	IBMzOS_ComputerSystem		
TotalAAPTimePercentage			
Description: zAAP time percentage.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% total logical utilization (AAP)	***DDS metric ID***	8D38D0
Associated class	IBMz_ComputerSystem		
TotalAAPonCPTimePercentage			
Description: zAAP on CP time percentage.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% AAP on CP	***DDS metric ID***	8D2C90
Associated class	IBMz_ComputerSystem		
TotalCPTimePercentage			
Description: Total CP time percentage.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% total logical utilization (CP)	***DDS metric ID***	8D2510
Associated class	IBMz_ComputerSystem		
TotalIIPonCPTimePercentage			
Description: zIIP on CP time percentage.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% IIP on CP	***DDS metric ID***	8D3550

Associated class	IBMz_ComputerSystem		
TotalIIPTimePercentage			
Description: Total zIIP time percentage.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% total logical utilization (IIP)	***DDS metric ID***	8D3900
Associated class	IBMz_ComputerSystem		

IBMzOS_OperatingSystem

This class contains basic properties of a running z/OS operating system image.

DelayForAAPPercentage (breakdown dimension: WLM service class period)			
Description: Percentage of samples where the reported WLM service class period has been found delayed for a zAAP.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of zAAP delay samples by WLM service class period	***DDS metric ID***	8D37E0
Associated class	IBMzOS_OperatingSystem		
DelayForCPPercentage (breakdown dimension: WLM service class period)			
Description: Percentage of samples where the reported WLM service class period has been found delayed for a standard processor.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of standard CP delay samples by WLM service class period	***DDS metric ID***	8D3740
Associated class	IBMzOS_OperatingSystem		
DelayForIIPPercentage (breakdown dimension: WLM service class period)			
Description: Percentage of samples where the reported WLM service class period has been found delayed for a zIIP.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of zIIP delay samples by WLM service class period	***DDS metric ID***	8D3830
Associated class	IBMzOS_OperatingSystem		
DelayForProcessorPercentage (breakdown dimension: WLM service class period)			

Description: Total percentage of samples where the reported WLM service class period has been found delayed for any used processor type.			
Datatype	real32	*Units*	percent
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	% of total delay samples by WLM service class period	*DDS metric ID*	8D3790
Associated class	IBMzOS_OperatingSystem		
DelayPercentage (breakdown dimension: WLM service class period)			
Description: Percentage of total delay samples.			
Datatype	real32	*Units*	percent
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	% delay by WLM service class period	*DDS metric ID*	8D17E0
Associated class	IBMzOS_OperatingSystem		
DelayCount (breakdown dimension: WLM service class period)			
Description: Number of samples where the reported WLM service class period has been found delayed.			
Datatype	real32	*Units*	samples
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	# of total delay samples by WLM service class period	*DDS metric ID*	8D43C0
Associated class	IBMzOS_OperatingSystem		
FreePhysicalMemory			
Description: Number of KBytes of physical memory currently unused and available. On z/OS, physical memory is also known as central storage.			
Datatype	uint64	*Units*	kilobytes
ChangeType	4 (Gauge)	*TimeScope*	2 (Point)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	# frames available	*DDS metric ID*	8D2EE0
Associated class	IBMzOS_OperatingSystem		
FreeSpaceInPagingFiles			
Description: Total number of free Kbytes in the operating system's paging files. For z/OS, this is the number of slots available in z/OS auxiliary storage.			
Datatype	uint64	*Units*	kilobytes
ChangeType	4 (Gauge)	*TimeScope*	2 (Point)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	# slots available	*DDS metric ID*	8D2F10

Associated class	IBMzOS_OperatingSystem		
FreeVirtualMemory			
Description: Number of KBytes of virtual memory currently unused and available.			
Datatype	uint64	***Units***	kilobytes
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# frames and slots available	***DDS metric ID***	8D2ED0
Associated class	IBMzOS_OperatingSystem		
InternalViewAAPTimePercentage			
Description: MVS view of the zAAP utilization.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% MVS utilization (zAAP)	***DDS metric ID***	8D3AE0
Associated class	IBMzOS_OperatingSystem		
InternalViewIIPTimePercentage			
Description: MVS view of the zIIP utilization.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% MVS utilization (zIIP)	***DDS metric ID***	8D3AF0
Associated class	IBMzOS_OperatingSystem		
InternalViewTotalCPUTimePercentage			
Description: MVS view of the CP utilization, only available for LPARs in which z/OS RMF is active.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% MVS utilization (CP)	***DDS metric ID***	8D0420
Associated class	IBMzOS_OperatingSystem		
KernelModeTime			
Description: Time in kernel mode on the operating system level. On z/OS, this is mapped to uncaptured time.			
Datatype	uint64	***Units***	milliseconds
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true

DDS metric name	uncaptured time	***DDS metric ID***	8D3240
Associated class	IBMzOS_OperatingSystem		
LoadAverage			
Description: Average in-ready queue length, that is, the number of workloads queued to be executed on a CP processor. In relation to the number of available CPs, this metric can be used as a CPU contention indicator, because workloads waiting in the in-ready queue are delayed for CPU resources.			
Datatype	real32	***Units***	queue length
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	load average	***DDS metric ID***	8D30E0
Associated class	IBMzOS_OperatingSystem		
LocalPI (breakdown dimension: WLM service class period)			
Description: Local performance index; the performance index of the MVS image on which the contacted CIM provider is running.			
Datatype	real32	***Units***	n/a
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	performance index by WLM service class period	***DDS metric ID***	8D1020
Associated class	IBMzOS_OperatingSystem		
NumberOfProcesses			
Description: Number of z/OS address spaces active during the sample interval. On other eServer systems, this is the number of processes on UNIX and the number of jobs on i5/OS.			
Datatype	uint32	***Units***	processes
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# users	***DDS metric ID***	8D0D50
Associated class	IBMzOS_OperatingSystem		
NumberOfUsers			
Description: Number of user sessions for which the operating system is currently storing state information. On z/OS, this is mapped to the number of TSO users currently active on the z/OS image.			
Datatype	uint32	***Units***	users
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	# users (Work scope: WLM Workload: TSO)	***DDS metric ID***	8D0D50

Associated class	IBMzOS_OperatingSystem		
OperationalStatus			
Description: Overall status of the associated CIM_OperatingSystem resource (= z/OS image). Currently, two states can be shown by the z/OS implementation: • 'Stressed' (= 0x10 as metric value), indicating that the system is severely overloaded • 'Ok' (= 0x04 as metric value). If the workflow with workscope SYSTEM is less than 60%, the system is reported to be 'Stressed', because in this case, even the workloads with highest priority are significantly delayed.			
Datatype	uint32	***Units***	n/a
ChangeType	n/a	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	false
DDS metric name	% workflow (based on DDS metric 0x8D0550 with work scope: WLM Workload: SYSTEM)	***DDS metric ID***	8D0550
Associated class	IBMzOS_OperatingSystem		
PageInRate			
Description: Number of pages paged in per second.			
Datatype	uint64	***Units***	events per second
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	pagein rate	***DDS metric ID***	8D30F0
Associated class	IBMzOS_OperatingSystem		
PercentDelay (z/OS only)			
Description: Percentage of samples during which the z/OS operating system was delayed for some resources.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% delay	***DDS metric ID***	8D0160
Associated class	IBMzOS_OperatingSystem		
RGCappingDelaySamples (breakdown dimension: WLM service class period)			
Description: Resource group capping delay samples percentage.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% of RG capping delay samples by WLM service class period	***DDS metric ID***	8D3880

Associated class	IBMzOS_OperatingSystem		
SRBTimePercentage (breakdown dimension: WLM service class period)			
Description: Percentage of SRB time used by all work in the system, or by WLM classes. This metric is divided by the number of processors. It does not include zAAP and zIIP times.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% SRB by WLM service class period	***DDS metric ID***	8D2D40
Associated class	IBMzOS_OperatingSystem		
SysplexPI (breakdown dimension: WLM service class period)			
Description: Sysplex performance index.			
Datatype	real32	***Units***	n/a
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	performance index by WLM service class period	***DDS metric ID***	8D1020
Associated class	IBMzOS_OperatingSystem		
TCBTimePercentage (breakdown dimension: WLM service class period)			
Description: Percentage of TCB time used by all work in the system, or by WLM classes. This metric is divided by the number of processors. It does not include zAAP and zIIP times.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% TCB by WLM service class period	***DDS metric ID***	8D2D50
Associated class	IBMzOS_OperatingSystem		
TotalAAPTimePercentage			
Description: zAAP utilization percentage			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% CPU utilization (zAAP)	***DDS metric ID***	8D39B0
Associated class	IBMzOS_OperatingSystem		
TotalAAPTimePercentage (breakdown dimension: WLM service class period)			
Description: zAAP utilization percentage for a given WLM service class period)			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)

GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	% zAAP	*DDS metric ID*	8D2C60
Associated class	IBMzOS_OperatingSystem		
TotalAAPonCPTimePercentage (breakdown dimension: WLM service class period)			
Description: Total zAAP on CP time percentage.			
Datatype	real32	*Units*	percent
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	% zAAP on CP by WLM service class period	*DDS metric ID*	8D2D00
Associated class	IBMzOS_OperatingSystem		
TotalCPUTime			
Description: Total system CPU time used.			
Datatype	uint64	*Units*	milliseconds
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	total time	*DDS metric ID*	8D31F0
Associated class	IBMzOS_OperatingSystem		
TotalIIPonCPTimePercentage (breakdown dimension: WLM service class period)			
Description: zIIP on CP time percentage.			
Datatype	real32	*Units*	percent
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	% zIIP on CP by WLM service class period	*DDS metric ID*	8D35D0
Associated class	IBMz_OperatingSystem		
TotalIIPTimePercentage			
Description: zIIP utilization percentage.			
Datatype	real32	*Units*	percent
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)
GatheringType	3 (Periodic)	*IsContinuous*	true
DDS metric name	% CPU utilization (zIIP)	*DDS metric ID*	8D39C0
Associated class	IBMzOS_OperatingSystem		
TotalIIPTimePercentage (breakdown dimension: WLM service class period)			
Description: zIIP utilization percentage for a given WLM service class period)			
Datatype	real32	*Units*	percent
ChangeType	4 (Gauge)	*TimeScope*	3 (Interval)

GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% CP utilization (zIIP)	***DDS metric ID***	8D3520
Associated class	IBMzOS_OperatingSystem		
UserModeTime			
Description: Time in user mode on operating system level. On z/OS, this is mapped to captured time.			
Datatype	uint64	***Units***	milliseconds
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	captured time	***DDS metric ID***	8D3030
Associated class	IBMzOS_OperatingSystem		

IBMzOS_ComputerSystem

This class provides basic computer system information such as computer name, and status information. It represents either virtual or physical computer systems in the sense of a container inside which an operating system runs. On zSeries, this class represents either an LPAR or a z/VM guest.

ActiveVirtualProcessors			
Description: Average number of regular CPs assigned to this IBMzOS_ComputerSystem (LPAR). If z/OS is running in a z/VM guest environment, this metric returns zero.			
Datatype	real32	***Units***	processors
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	average number of logical processors	***DDS metric ID***	8D2610
Associated class	IBMzOS_ComputerSystem		
PartitionDefinedCapacityUsedPercentage			
Description: Percentage of defined capacity actually used by the partition. If z/OS is running in a z/VM guest environment, this metric returns zero.			
Datatype	uint32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% capacity used	***DDS metric ID***	8D2870
Associated class	IBMzOS_ComputerSystem		

IBMzOS_Process

This class provides basic process information such as process name, priority, and run-time state. Instances of class IBMzOS_Process are mapped to z/OS address spaces. Client applications can use this class to give clients an understanding of the processes (address spaces) running on the managed system within the context of their operating system.

KernelModeTime			
Description: Time in kernel mode, in milliseconds. This metric is an eServer metric and currently not implemented for z/OS.			
Datatype	uint64	***Units***	milliseconds
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	not implemented	***DDS metric ID***	not implemented
Associated class	IBMzOS_Process		
PageInRate			
Description: Number of pages paged in per second on behalf of the associated process.			
Datatype	uint64	***Units***	kilobytes
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	pagin rate per residency time by job	***DDS metric ID***	8D1090
Associated class	IBMzOS_Process		
ResidentSetSize			
Description: Memory in bytes currently allocated in physical memory by the given process. Also known as 'working set size'.			
Datatype	uint64	***Units***	bytes
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	working set by job	***DDS metric ID***	8D1280
Associated class	IBMzOS_Process		
TotalCPUTime			
Description: Amount of CPU time used by the given process.			
Datatype	uint64	***Units***	milliseconds
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	eappl time by job	***DDS metric ID***	8D3070
Associated class	IBMzOS_Process		
UserModeTime			

Description: CPU time used in user mode specifically for the given process. On z/OS this is currently the same as **TotalCPUTime**.			
Datatype	uint64	***Units***	milliseconds
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	eappl time by job	***DDS metric ID***	8D3070
Associated class	IBMzOS_Process		

IBMzOS_Processor

This class represents the physical processors that are available to the z/OS operating system.

TotalCPUTimePercentage			
Description: Percentage of time the CPU was not idle. This metric is an eServer metric and currently not implemented for z/OS.			
Datatype	uint8	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	not implemented	***DDS metric ID***	not implemented
Associated class	IBMzOS_Processor		

IBMzOS_UnixProcess

This class provides basic information about z/OS processes running in the UNIX System Services (USS) subsystem. It implements all properties from CIM_Process plus a set of properties typical for Unix processes.

AccumulatedTotalCPUTime			
Description: CPU time in milliseconds spent for this process since USS process creation.			
Datatype	uint64	***Units***	milliseconds
ChangeType	3 (Counter)	***TimeScope***	4 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	total cpu seconds by uss pid and jobname	***DDS metric ID***	8D31E0
Associated class	IBMzOS_UnixProcess		
ExternalViewUserModePercentage			
Description: Usage percentage of the system CPUs for this process in user mode during the measurement interval.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% appl by uss pid and jobname	***DDS metric ID***	8D2830
Associated class	IBMzOS_UnixProcess		

IBMzOS_Channel

Instances of this class represent z/OS channels, with property DeviceID containing the Channel Path ID (CHPID). The purpose of this resource class is to enable the association of related metrics.

Note: Channels are specific for zSeries. Therefore, the provided metrics are not platform independent eServer metrics.

BusUtilization			
Description: Percentage of bus cycles, the bus has been found busy for this channel in relation to the theoretical limit.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% bus utilization	***DDS metric ID***	8D2360
Associated class	IBMzOS_Channel		
BytesReceived			
Description: Total number of bytes received per second, including framing characters			
Datatype	uint64	***Units***	bytes
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	total bytes read/sec	***DDS metric ID***	8D23D0
Associated class	IBMzOS_Channel		
BytesTransmitted			
Description: Total number of bytes transmitted per second, including framing characters.			
Datatype	uint64	***Units***	bytes
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	total bytes written/sec	***DDS metric ID***	8D23F0
Associated class	IBMzOS_Channel		
ErrorRate			
Description: Number of network errors per second.			
Datatype	real32	***Units***	errors per second
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	send fail/sec	***DDS metric ID***	8D31C0
Associated class	IBMzOS_Channel		
NetworkPortUtilizationPercentage			
Description: Percentage of z/OS channel capacity actually used. This number may not be precise, meaning that for some non-ideal work loads (like transferring lots of very short messages), it may not be possible to reach 100% utilization.			
Datatype	uint32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true

DDS metric name	% total utilization	***DDS metric ID***	8D0080
Associated class	IBMzOS_Channel		
PartitionBytesReceived (z/OS only)			
Description: Total number of bytes received.			
Datatype	uint64	***Units***	bytes
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	partition bytes read/sec	***DDS metric ID***	8D2390
Associated class	IBMzOS_Channel		
PartitionBytesTransmitted (z/OS only)			
Description: Total number of bytes transmitted.			
Datatype	uint64	***Units***	bytes
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	partition bytes written/sec	***DDS metric ID***	8D23B0
Associated class	IBMzOS_Channel		
PartitionUtilization			
Description: Channel path utilization percentage for an individual logical partition.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% partition utilization	***DDS metric ID***	8D0060
Associated class	IBMzOS_Channel		
ReceiveErrorRate (z/OS only)			
Description: Number of network errors per second related to receiving activities of the channel.			
Datatype	real32	***Units***	errors per second
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)
GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	receive fail/sec	***DDS metric ID***	8D3160
Associated class	IBMzOS_Channel		
TotalUtilization			
Description: Channel path utilization percentage for the entire system during the interval.			
Datatype	real32	***Units***	percent
ChangeType	4 (Gauge)	***TimeScope***	3 (Interval)

GatheringType	3 (Periodic)	***IsContinuous***	true
DDS metric name	% total utilization	***DDS metric ID***	8D0080
Associated class	IBMzOS_Channel		

IBMzOS_UnixLocalFileSystem

This class represents Unix file systems that are locally attached to a computer system. On z/OS, hierarchical file system ZFS is supported.

Note: The class IBMzOS_UnixLocalFileSystem is inherited from CIM_FileSystem.

AvailableSpace			
Description: Total amount of free space for the associated file system in bytes. This metric is not implemented based on RMF data.			
Datatype	uint64	***Units***	bytes
ChangeType	4 (Gauge)	***TimeScope***	2 (Point)
GatheringType	4 (OnRequest)	***IsContinuous***	true
DDS metric name	none	***DDS metric ID***	none
Associated class	IBMzOS_UnixLocalFileSystem		

CIM indication support

RMF can generate indications for monitoring data and thus enable CIM clients to support event-based monitoring. A CIM client can subscribe for certain events. While a subscription is active, the RMF indication provider checks the condition autonomously and, if necessary, delivers a CIM indication to notify the subscribed CIM client(s) that the condition became true or the event occurred.

Introduction to CIM indications

CIM indications are supported with the port of the OpenPegasus 2.5 CIM server to z/OS V1R8. Typical CIM indications in an RMF context are notifications that some metric value has exceeded a given threshold.

To specify an event about which you want to be notified, you construct appropriate filter strings using an SQL like query language modified for CIM requirements. This language is called ***CIM Query Language***, abbreviated as ***CQL***. You can download the CQL specification from the ***DMTF web page*** (http://www.dmtf.org). The DMTF document number is DSP0202.

With this support, CIM clients can subscribe to certain indications which are of interest for them and thus are notified by the destination CIM server if such an indication occurs. Here are two examples for indication subscriptions that you can implement based on the RMF CIM indication provider:

Example 1: You can subscribe for an indication whenever the `LoadAverage` metric for ***IBMzOS_OperatingSystem*** is above 20. This metric indicates the average length of the in-ready queue of currently dispatchable units. From your experience with your typical workload and environment, you may know that a value greater than 20 could indicate a CPU contention.

Example 2: You want to receive a notification whenever any address space has a working set size of more than 512 MB. Or you want to be notified whenever any channel is more than 80% utilized. This is conceptually different from **Example 1**, which demonstrates the use of a single metric instance associated to a single resource, whereas **Example 2** handles a complete set of metric instances with the same associated metric definition.

A common problem of indication infrastructures is that they can consume a significant amount of resources if too many subscriptions are active. However, you can most probably avoid such a situation by using the RMF default settings of the provided environment variables as described in “RMF indication polling” on page 129.

You can find a general introduction to the CIM indication support in the *z/OS Common Information Model User’s Guide* (SC33-7998). Detailed information about CIM indications is in the ***CIM Indications (Events) White Paper, DSP0107***.

How to subscribe for RMF CIM indications

To receive a notification from the CIM indication provider, you must subscribe to the appropriate CIM server to which this provider is registered. For this purpose, you need to create an ***IndicationSubscription*** instance with associated instances of **CIM_IndicationFilter** and **CIM_ListenerDestination** in the **PG_InterOp** namespace. To create such an instance, follow these steps:

1. Get the object path of the metric instance (see “How to monitor a certain address space” on page 117) or the ***CIM_BaseMetricDefinition.Id*** property of the used metric (see “How to monitor several address spaces for a certain condition” on page 118) to be used in the ***CIM_IndicationFilter*** instance in step 2.
2. Create a ***CIM_IndicationFilter*** instance with a query filter, defining a desired set of indications (see “How to create a CIM_IndicationFilter instance”). In particular, the ***CIM_IndicationFilter*** instance defines the namespace, indication class, filter criteria and data project list of the desired indication stream. The ***CIM_IndicationFilter*** instance also contains filter strings in CQL.
3. Create a ***CIM_ListenerDestinationCIMXML*** instance which describes where the indication should be delivered (TCP/IP address of the CIM client where the CIM listener thread has been started).
4. Create a ***CIM_IndicationSubscription*** instance, which combines the ***CIM_IndicationFilter*** and the ***CIM_ListenerDestinationCIMXML*** instances.
5. Now, an indication listener (subscriber) at the destination address may receive indications.

How to create a CIM_IndicationFilter instance

The following sample code shows a query string for an indication filter used in the scenario described in “How to monitor a certain address space” on page 117. Those parts that typically remain the same for any filter are displayed **in bold**, while those that vary depending on what you want to monitor are displayed *in italic*.

```
SELECT *
  FROM CIM_InstModification
  WHERE SourceInstance ISA IBMzOS_BaseMetricValue
    AND SourceInstance.IBMzOS_BaseMetricValue::InstanceID=
```

```
    '#presource=SYSE,*,
    CENTRAL_STORAGE#aid=8D1280#afilter=HI=99999;PAT=MVSNFSS#rASID(00B9)'
AND stringtouint(SourceInstance.IBMzOS_BaseMetricValue::MetricValue)
 < 10000000
```

The object path of the metric value - in our example: `"#presource=SYSE,*, CENTRAL_STORAGE#aid=8D1280#afilter=HI=99999;PAT=MVSNFSS#rASID(00B9)"` - has been identified in step 1 on page 116.

You need the ***stringtouint*** function to convert the value of a metric that is represented as a string within an instance into an integer value that you can compare to your desired threshold values. To compare metric values, you can use the common operators which are part of the CQL standard, for example, `<=` denotes `'less than or equal'` or `<>` denotes `'not equal'`.

How to monitor a certain address space

Consider the following scenario: You want to generate a CIM indication, whenever the `LoadAverage` metric - that is the average in-ready queue length - of your z/OS MVS image is greater than 20.

Perform the following steps:

1. Get the ***CIM_BaseMetricDefinition*** instance where the ***CIM_BaseMetricDefinition.Name*** property is equal to "`LoadAverage`". This metric instance has an ID of `8D30E0`.

   You achieve this using a z/OS UNIX command line utility: You can have a look at the output of the following command:

   ```
   cimcli ei CIM_BaseMetricDefinition -niq
   ```

   The output of such a command looks similar to the following:

   ```
   instance of class IBMzOS_BaseMetricDefinition
   {
       ...
       string ElementName = "z/OS load average; average in-ready queue length"
                            "including enclaves, threads, USS processes, etc.";
       string Id = "8D30E0";
       string Name = "LoadAverage";
       ...
       string Units = "QueueLength";
       ...
   };
   ```

   The only key property (as defined in the MOF and also visible from CIM client applications) is the "Id" property (with value 8D30E0 in the example), so the CIM object path of this metric is

   ```
   IBMzOS_BaseMetricDefinition.Id="8D30E0"
   ```

2. Traverse the ***CIM_MetricInstance*** association from the `LoadAverage` metric instance to find all object paths of `LoadAverage` metrics. You can achieve this with a z/OS UNIX command like:

   ```
   cimcli an 'IBMzOS_BaseMetricDefinition.Id="8D30E0"' -ac CIM_MetricInstance"
   ```

   which will return something like this:

   ```
   //BOESYSE/root/cimv2:IBMzOS_BaseMetricValue.InstanceID="#presource=SYSE,
                         *,PROCESSOR#aid=8D30E0"
   ```

3. Create a ***CIM_IndicationFilter*** instance: in our example, we want to receive indications whenever the `LoadAverage` of the running MVS image is above 20. This is an example of an indication subscription that is associated to a metric value of a certain resource, that is, a "GetInstance"-based subscription.

```
SELECT *
  FROM CIM_InstModification
  WHERE SourceInstance ISA IBMzOS_BaseMetricValue
    AND SourceInstance.IBMzOS_BaseMetricValue::InstanceID=
        '#presource=SYSE,*,PROCESSOR#aid=8D30E0'
    AND stringtouint(SourceInstance.IBMzOS_BaseMetricValue::MetricValue)
        > 20
```

4. Create a ***CIM_ListenerDestinationCIMXML*** instance. For this purpose, you can use a program as shown in "Sample program to subscribe for and to receive indications."
5. Create a ***CIM_IndicationSubscription*** instance. For this purpose, you can also use the program similar to the one shown in "Sample program to subscribe for and to receive indications."

How to monitor several address spaces for a certain condition

In this scenario, you want to generate a CIM indication, whenever the *ResidentSetSize* (*WorkingSetSize*) of any address space becomes greater than 2 GB, that means of all currently active address spaces in the image represented by an instance of ***CIM_OperatingSystem***.

1. The first step in this scenario is similar to step 1 on page 117 of the scenario described in "How to monitor a certain address space". As a difference, in this case, you just need to look what is the value of the property ***Id*** of the ***CIM_BaseMetricDefinition*** instance with the ***Name*** property equal to "ResidentSetSize". In the example, the ***Id*** value of that metric definition instance is 8D1280.
2. In this example, we filter for any address spaces with a resident set size greater than 2 GB (2 x 1024 x 1024 x 1024 = 2.147.483.648 byte).

```
SELECT *
  FROM CIM_InstModification
  WHERE SourceInstance ISA IBMzOS_BaseMetricValue
  AND SourceInstance.IBMzOS_BaseMetricValue::MetricDefinitionId ='8D1280'
  AND stringtouint(SourceInstance.IBMzOS_BaseMetricValue::MetricValue)
  > 2147483648
```

Sample program to subscribe for and to receive indications

You can use the following sample Java client to subscribe for and to receive indications. To activate the subscription from the scenario "How to monitor a certain address space" on page 117, you can enter the following:

```
java RMFIndicationClient <mvs_hostname> <cimserver_port>
<tso_userid> <tso_password> <destination_port>
create_subscription <subscr_name>
"SELECT *
 FROM CIM_InstModification
 WHERE SourceInstance ISA IBMzOS_BaseMetricValue
 AND SourceInstance.IBMzOS_BaseMetricValue::InstanceID=
     'IBMzOS_BaseMetricValue.InstanceID="#presource=SYSE,*,
     PROCESSOR#aid=8D30E0'
 AND stringtouint(SourceInstance.IBMzOS_BaseMetricValue::MetricValue) > 20"
```

where:

mvs_hostname	TCP/IP name of the MVS image on which the CIM server is running
cimserver_port	TCP/IP CIM Server port (default 5988)
destination_port	TCP/IP port of the CIM listener
subscr_name	self chosen name for the subscription which must be unique within the CIM server

Likewise, to create a subscription with the filter used in scenario “How to monitor several address spaces for a certain condition” on page 118, you specify a command similar to the following, that invokes the sample program (see below) together with the filter string as input parameter:

```
java RMFIndicationClient <mvs_hostname> <cimserver_port>
<tso_userid> <tso_password> <destination_port>
create_subscription <subscr_name>
"SELECT *
FROM CIM_InstModification
WHERE SourceInstance ISA IBMzOS_BaseMetricValue
AND SourceInstance.IBMzOS_BaseMetricValue::MetricDefinitionId='8D1280'
AND stringtouint(SourceInstance.IBMzOS_BaseMetricValue::MetricValue) > 2147483648"
```

Note: The destination TCP/IP address for this Java test client program is automatically the system from which the request to create a ***CIM_IndicationSubscription*** instance was issued, for example the Linux or Windows system on your desktop running the SBLIM Java client with the sample program.

Sample program:

```
/**
 * Module name: RMFIndicationClient.java
 * (programming language Java)
 *
 * Licensed Materials - Property of IBM
 * "Restricted Materials of IBM"
 * (C) Copyright IBM Corp. 2006
 * Status=HRM7730 (5694-A01 z/OS V1.8 RMF)
 *
 * This sample Java client can be used
 * to subscribe for indications and to
 * receive indications.
 *
 * Build:
 * - install IBM Java 1.4.2
 * - install the sblim-cim-client package
 *   from http://sblim.sourceforge.net/
 * - compile this program with "javac RMFIndicationClient.java"
 *
 * Usage:
 * - after starting this program
 *   ("java RMFIndicationClient"),
 *   this program will display a short help text
 * - note that query filters have to
 *   use the DMTF:CQL query language
 *
 * If you have any questions, please
 * write an Email to rmf@de.ibm.com
 */

// generic Java includes
import java.net.*;
import java.util.*;

// SBLIM CIM Client includes
import org.sblim.wbem.cim.*;
import org.sblim.wbem.client.*;
import org.sblim.wbem.client.indications.*;
import org.sblim.wbem.http.*;


public class RMFIndicationClient {

    // global variables for data read from
    // the command line
    private static String user;       // TSO user ID
    private static String pwd;        // TSO password
    private static String host;       // MVS host name
    private static String portNumber; // OpenPegasus port number, like 5988
    private static int listenPort;    // Indication listen port number, like 2003
    private static String command;    // command (see help)
    private static String query;      // filter string in query language DMTF:CQL
    private static String subscr_name;// unique name of the subscription

    private static CIMClient cimClient;  // CIM client instance for connection to the OpenPegasus
                                         // server
```

```
   /**
    * print out a help text if invalid commands were entered
    */
   private static void printHelpText() {
    System.out.println(
       "Usage: RMFIndicationClient <host> <port> <user> <password> <listen_port> \n" +
       "               create_subscription <subscription_name> <CQL query>\n" +
       "             | remove_subscription <subscription_name> <CQL query>\n" +
       "             | list_subscriptions\n" +
       "             | remove_all_subscriptions\n" +
       "             | start_listener\n");
   }

   /**
    * establish a connection to the z/OS OpenPegasus server
    */
   private static void connectToServer() {
    UserPrincipal userPr = new UserPrincipal(user);
PasswordCredential pwCred = new PasswordCredential(pwd.toCharArray());
CIMNameSpace ns = new CIMNameSpace(
       host.startsWith("http://")
       ? (host+":"+portNumber)
       : "http://" + host+":"+portNumber,
       "root/PG_InterOp");

System.out.println("connectToServer("
     + host + ":" + portNumber + ") called");

cimClient = new CIMClient(ns,userPr,pwCred);
cimClient.setLocale(new Locale("en","US"));
   }

   /**
    * ask OpenPegasus for MVS system name
    */
   private static String getSystemNameFromServer() {
String systemName = "";

CIMObjectPath objectPath = new CIMObjectPath("IBMzOS_ComputerSystem");
objectPath.setNameSpace("root/cimv2");

Enumeration systemInstanceNames =
    cimClient.enumerateInstanceNames(objectPath);

if (systemInstanceNames.hasMoreElements()) {

    CIMObjectPath systemInstanceName = (CIMObjectPath) systemInstanceNames.nextElement();
    systemName = systemInstanceName.getKey("Name").getValue().getValue().toString();
}

return systemName;
   }
```

```
/**
 * create a new CIM indication subscription
 */
private static void createSubscription( CIMClient cimClient, String destination ) {
try {
   String systemName = getSystemNameFromServer();

   System.out.println( "system name " + systemName + "   destination " + destination );
   // create CIM_ListenerDestinationCIMXML instance
   CIMClass indicationDestinationClass =
cimClient.getClass(new CIMObjectPath("CIM_ListenerDestinationCIMXML"), false);
   CIMInstance indicationDestinationInstance = indicationDestinationClass.newInstance();

   indicationDestinationInstance.
setProperty(
     "CreationClassName",
     new CIMValue("CIM_ListenerDestinationCIMXML",
    CIMDataType.getPredefinedType(CIMDataType.STRING)));
   indicationDestinationInstance.
setProperty( "Name",
      new CIMValue( "IBMzOS_RMF_" + subscr_name + "_Destination",
      CIMDataType.getPredefinedType(CIMDataType.STRING)));

   indicationDestinationInstance.
setProperty(
     "SystemCreationClassName",
     new CIMValue("CIM_ComputerSystem",
    CIMDataType.getPredefinedType(CIMDataType.STRING)));
   indicationDestinationInstance.
setProperty( "SystemName",
      new CIMValue(systemName,
     CIMDataType.getPredefinedType(CIMDataType.STRING)));

   // Set PersistenceType to 3 (transient), destination is shortlived, affects errorhandling
   indicationDestinationInstance.setProperty("PersistenceType",
          new CIMValue(new UnsignedInt16(3),
         CIMDataType.
         getPredefinedType(
          CIMDataType.UINT16)));
   indicationDestinationInstance.setProperty("Destination",
          new CIMValue(destination,
         CIMDataType.getPredefinedType(
        CIMDataType.STRING)));

   CIMObjectPath indicationDestinationPath
= cimClient.createInstance(new CIMObjectPath(),
      indicationDestinationInstance);

   // create CIM_IndicationFilter instance
   CIMClass indicationFilterClass
= cimClient.getClass(
       new CIMObjectPath("CIM_IndicationFilter"), false);
   CIMInstance indicationFilterInstance = indicationFilterClass.newInstance();

   indicationFilterInstance.
setProperty("CreationClassName",
     new CIMValue("CIM_IndicationFilter",
    CIMDataType.getPredefinedType(CIMDataType.STRING)));
   indicationFilterInstance.
setProperty("Name",
     new CIMValue( "IBMzOS_RMF_" + subscr_name + "_Filter",
     CIMDataType.getPredefinedType(CIMDataType.STRING)));
```

```
    indicationFilterInstance.
 setProperty("SystemCreationClassName",
      new CIMValue("CIM_ComputerSystem",
     CIMDataType.getPredefinedType(CIMDataType.STRING)));
    indicationFilterInstance.
 setProperty("SystemName",
      new CIMValue(systemName,
     CIMDataType.getPredefinedType(CIMDataType.STRING)));

    indicationFilterInstance.
 setProperty("Query",
      new CIMValue(query,
     CIMDataType.getPredefinedType(CIMDataType.STRING)));
    indicationFilterInstance.
 setProperty("QueryLanguage",
      new CIMValue("DMTF:CQL",
     CIMDataType.getPredefinedType(CIMDataType.STRING)));
    indicationFilterInstance.
 setProperty("SourceNamespace",
      new CIMValue("root/cimv2",
     CIMDataType.getPredefinedType(CIMDataType.STRING)));

    CIMObjectPath indicationFilterPath
 = cimClient.createInstance(new CIMObjectPath(),
       indicationFilterInstance);

    // create CIM_IndicationSubscription instance
    CIMClass indicationSubscriptionClass
 = cimClient.getClass(
    new CIMObjectPath("CIM_IndicationSubscription"), false);
    CIMInstance indicationSubscriptionInstance
 = indicationSubscriptionClass.newInstance();
    indicationSubscriptionInstance.
 setProperty("Filter",
      new CIMValue(indicationFilterPath,
     CIMDataType.getPredefinedType(CIMDataType.REFERENCE)));
    indicationSubscriptionInstance.
 setProperty("Handler",
      new CIMValue(indicationDestinationPath,
     CIMDataType.getPredefinedType(CIMDataType.REFERENCE)));
    indicationSubscriptionInstance.
 setProperty("SubscriptionState",
      new CIMValue(new UnsignedInt16(2),
     CIMDataType.getPredefinedType(CIMDataType.UINT16)));

    CIMObjectPath indicationSubscriptionPath
 = cimClient.createInstance( new CIMObjectPath(),
        indicationSubscriptionInstance );

    System.out.println("... subscription successfully created!");
} catch (CIMException e) {
    if (CIMException.CIM_ERR_ALREADY_EXISTS.equals(e.getID())) {
 System.out.println("Instances already exist on server.");
    } else {
 e.printStackTrace();
    }
}
   }
```

```
   /**
    * list all subscriptions currently active on the CIM server
    */
   private static void listSubscriptions() {
CIMObjectPath cop = new CIMObjectPath("CIM_IndicationFilter");
Enumeration e = cimClient.enumerateInstances(cop);
while ( e.hasMoreElements() ) {
    CIMInstance ci = (CIMInstance)e.nextElement();
    System.out.println( ci.getProperty("Query") + "  " + ci.getProperty("Name") );
}
   }

   /**
    * remove all subscriptions
    */
   private static void removeSubscriptions() {
CIMObjectPath cop = new CIMObjectPath("CIM_IndicationSubscription");
Enumeration e = cimClient.enumerateInstances(cop);
CIMInstance inst = null;

while ( e.hasMoreElements() ) {
    inst = (CIMInstance)e.nextElement();
    System.out.println( "deleting " + inst.getObjectPath() );
    cimClient.deleteInstance( inst.getObjectPath() );
}

cop = new CIMObjectPath("CIM_IndicationFilter");
e = cimClient.enumerateInstances(cop);
while ( e.hasMoreElements() ) {
    inst = (CIMInstance)e.nextElement();
    System.out.println( "deleting " + inst.getObjectPath() );
    cimClient.deleteInstance( inst.getObjectPath() );
}

cop = new CIMObjectPath("CIM_ListenerDestinationCIMXML");
e = cimClient.enumerateInstances(cop);
while ( e.hasMoreElements() ) {
    inst = (CIMInstance)e.nextElement();
    System.out.println( "deleting " + inst.getObjectPath() );
    cimClient.deleteInstance( inst.getObjectPath() );
}

   }
```

```
   /**
    * remove a single subscription, identified by filterName
    */
   private static void removeSubscription() {

String filterName = "IBMzOS_RMF_" + subscr_name + "_Filter";
String handlerName = "IBMzOS_RMF_" + subscr_name + "_Destination";

System.out.println("try to remove " + filterName);

CIMInstance filterInstance = null;
CIMInstance handlerInstance = null;
CIMInstance subscrInstance = null;

CIMObjectPath cop = new CIMObjectPath("CIM_IndicationFilter");
Enumeration e = cimClient.enumerateInstances(cop);
boolean found = false;
while ( e.hasMoreElements() && ! found ) {
    filterInstance = (CIMInstance)e.nextElement();
    String xFilterName = filterInstance.getProperty("Name").getValue().getValue().toString();
    System.out.println("filter name : " + xFilterName );
    if ( xFilterName.equals( filterName ) ) {
 found = true;
 System.out.println("found!");
    }

}
if ( found ) {
    System.out.println("try to remove " + handlerName );
    found = false;
    cop = new CIMObjectPath("CIM_ListenerDestinationCIMXML");
    e = cimClient.enumerateInstances(cop);
    while ( e.hasMoreElements() && ! found ) {
 handlerInstance = (CIMInstance)e.nextElement();
 String xHandlerName
     = handlerInstance.getProperty("Name").getValue().getValue().toString();
 System.out.println("handler name: " + xHandlerName );
 if ( xHandlerName.equals( handlerName ) ) {
     System.out.println("found handler!");
     found = true;
 }
    }
}
if ( found ) {
    cop = new CIMObjectPath("CIM_IndicationSubscription");
    e = cimClient.enumerateInstances(cop);
    found = false;
    while ( e.hasMoreElements() && ! found ) {
 subscrInstance = (CIMInstance)e.nextElement();
 if ( subscrInstance.getProperty("Filter").
      getValue().getValue().toString().
      equals( filterInstance.getObjectPath().toString() )
      && subscrInstance.getProperty("Handler").
      getValue().getValue().toString().
      equals(  handlerInstance.getObjectPath().toString() ) ) {
     found = true;
     System.out.println( "found subscription!" );
 }

    }
}
```

```
if ( ! found ) {
    System.out.println("Subscription not found");
} else {
    System.out.println( "deleting " + subscrInstance.getObjectPath() );
    cimClient.deleteInstance( subscrInstance.getObjectPath() );
    System.out.println( "deleting " + filterInstance.getObjectPath() );
    cimClient.deleteInstance( filterInstance.getObjectPath() );
    System.out.println( "deleting " + handlerInstance.getObjectPath() );
    cimClient.deleteInstance( handlerInstance.getObjectPath() );

}
   }

   /*** main program  ***/

   public static void main(String[] args) {
if (args.length > 8 || args.length < 5) {
    System.out.println("Error: Not a valid number of arguments! - "+args.length);
    System.out.println();
    printHelpText();
} else {
    host = args[0];
    portNumber  = args[1];
    user = args[2];
    pwd  = args[3];
    if ( args.length > 4 ) {
 listenPort = Integer.parseInt(args[4]);
 if ( args.length > 5 ) {
     command = args[5];
     if ( args.length > 6 ) {
  subscr_name = args[6];
  if ( args.length > 7 ) {
      query = args[7];
  }
     }
 }
    }

    System.out.println("z/OS RMF CIM Indication client");

    if  (command.equals("create_subscription")) {
 System.out.println("create subscription");
 connectToServer();
 try {
     String ident = "";
     String destination="";
     destination = "http://" + InetAddress.getLocalHost().getHostAddress()
  + ":" + listenPort + "/" + ident;
     createSubscription( cimClient, destination );
 } catch (UnknownHostException e) {
     System.out.println("Error: Could not determine address of localhost!");
 } catch (NumberFormatException e) {
     System.out.println("Error: Parameter listenerport is not a numerical value!");
 }
    } else if  (command.equals("start_listener")) {
 System.out.println("start listener");
 connectToServer();
 RMFConsumerThread consumer
     = new RMFConsumerThread( listenPort );
 consumer.run();
    } else if ( command.equals("remove_all_subscriptions") ) {
 System.out.println("remove subscriptions");
 connectToServer();
 removeSubscriptions();
```

```
      } else if ( command.equals("list_subscriptions") ) {
  System.out.println("list subscriptions");
  connectToServer();
  listSubscriptions();
      } else if ( command.equals("remove_subscription") ) {
  System.out.println("remove subscription");
  connectToServer();
  removeSubscription();
      } else {
  System.out.println("Error: Command not known!");
  System.out.println();
  printHelpText();
      }

      if ( null != cimClient ) {
  cimClient.close();
      }
 }
    }
/**
 * Module name: RMFConsumerThread.java (programming language Java)
 *
 * Licensed Materials - Property of IBM
 * "Restricted Materials of IBM"
 * (C) Copyright IBM Corp. 2006
 * Status=HRM7730 (5694-A01 z/OS V1.8 RMF)
 *
 * This class is used by the RMF sample program module
 * RMFIndicationClient.java
 * If you have any questions, please write an Email to rmf@de.ibm.com
 */

import java.net.*;
import java.util.*;
import org.sblim.wbem.cim.*;
import org.sblim.wbem.client.*;
import org.sblim.wbem.client.indications.*;
import org.sblim.wbem.http.*;

public class RMFConsumerThread extends Thread {
    private static int iv_listenPort;

    public RMFConsumerThread( int listenPort ) {
 iv_listenPort = listenPort;
    }

    public static void createIndicationListener() {
 RMFListener rmfListener =
     new RMFListener();

 // All listeners should be registered in a CIMIndicationListenertList
 CIMIndicationListenertList indicationClient
     = new CIMIndicationListenertList();

 indicationClient.addListener(rmfListener);
 // CIMEventDispatcher dispatches the events to the registered listeners in
 // the CIMIndicationListenertList
 CIMEventDispatcher dispatcher
     = new CIMEventDispatcher(indicationClient);
```

```
 // CIMIndicationHandler is parsing the received indication messages
 CIMIndicationHandler indicationHdlr
     = new CIMIndicationHandler(dispatcher);

 HttpServerConnection indicationServer = null;
 try {
     indicationServer
  = new HttpServerConnection(
         new HttpConnectionHandler(indicationHdlr),
         iv_listenPort);
     indicationServer.setName("RMF CIM indication listener");
     indicationServer.start();
     while (true) {
  System.out.println("waiting for indications ...");
  Thread.sleep(10000);
     }
 } catch (Exception e) {
     dispatcher.kill();
     e.printStackTrace();
 }

 System.out.println("indication listener created");
    }

    public synchronized void run() {
 createIndicationListener();
 try {
     wait();
 } catch( InterruptedException e ) {
     e.printStackTrace();
 }
    }
}

 /**
 * Module name: RMFListener.java
 * (programming language Java)
 *
 * Licensed Materials - Property of IBM
 * "Restricted Materials of IBM"
 * (C) Copyright IBM Corp. 2006
 * Status=HRM7730 (5694-A01 z/OS V1.8 RMF)
 *
 * This class is used by the RMF sample program module
 * RMFIndicationClient.java
 *
 * If you have any questions, please
 * write an Email to rmf@de.ibm.com
 */

import org.sblim.wbem.cim.*;
import org.sblim.wbem.client.indications.*;

public class RMFListener implements CIMListener {

 public void indicationOccured(CIMEvent e) {

  CIMInstance indication = e.getIndication();
  System.out.println("caught CIM indication: " + indication);
 }

}
```

RMF indication polling

RMF's design goal was not to consume too many CPU resources with checking the indications. As only one dedicated z/OS UNIX thread is used for indication checking, the indication provider cannot consume more than 1 CPU. In addition, there are the following environment variables with which you can influence the workload:

- With the environment variable `RMF_INDICATION_POLLING_INTERVAL`, you can control the frequency how often every single subscription is checked (in seconds). For example, if `RMF_INDICATION_POLLING_INTERVAL` is set to 300, every subscription is checked only once within 5 minutes. The default value is 100 seconds, synchronously with the default RMF Monitor III MINTIME.
- You can set `RMF_INDICATION_RESTTIME` to specify how many seconds the provider should sleep between two indication subscriptions are checked. The default is one second.

As an example, if you have three active CIM clients (also called ***indication subscriber*** in this case), which want to be notified whenever any address space consumes more than 1 GB of working set size and another active indication subscriber which wants to be notified whenever the CIM server address space consumes more than 10% CPU, the following happens in a loop:

1. The RMF indication provider retrieves all working set sizes of address spaces and compares them with the threshold. If the threshold condition evaluates to TRUE, an indication is delivered. Now the z/OS CIM server identifies which clients have subscribed/registered for this indication and delivers the indication to the three subscribers.
2. Let us assume, the RMF default of RMF_INDICATION_RESTTIME is used. In this case, the RMF indication provider sleeps for one second before checking the next subscription.
3. The subscription for the single address space with more than 10% CPU consumption is evaluated with a similar processing as described in step 1.
4. Again, the RMF indication provider will sleep for one second, like in step 2.
5. In this example, all indications have been checked within two seconds. The indication provider can sleep for 98 seconds in order to avoid checking the same subscription twice within one RMF Monitor III MINTIME interval.

Notes:

1. There is a conflict between resource consumption and reaction to events: to reduce the resource consumption caused by the indication delivery logic, you may want to provide high values for the above environment variables. If you have a very high number of active subscriptions and you want to have them checked frequently, you may try to reduce `RMF_INDICATION_RESTTIME` to zero. However, in this case, RMF may consume up to slightly more than one CPU of your system for indication checking.
2. The resource consumption rather depends on the number of different indications checked, than on the number of active indication subscribers which would like to receive indications. So if you deliver the same indication to a high number of clients, this should not have too much impact on system performance.

How and where to set the mentioned environment variables is described in the *z/OS Common Information Model User's Guide*.

Chapter 5. Adding Monitor I and Monitor II installation exits

This chapter describes:
- How to create Monitor I user exit routines
- How to create Monitor II user reports

Overview

Facilities in RMF allow you to gather and report data relevant to your installation.

During a Monitor I session, installation exits let you sample data at each RMF cycle, collect this data and examine system indicators at each RMF interval, format and write your own SMF records, and format and write your own reports. You can also use the RMF trace facilities to trace the contents of any SQA, fixed CSA, or nucleus field that you require. During a Monitor II session, the data interface service allows you to directly access SMF record data from storage in real time rather than through SMF. The service provides easy access to this data by invoking the module ERBSMFI.

During a Monitor II session, installation exits enable you to gather and report your own data by coding your own data-gathering and data-reporting routines. RMF provides the USER option for a background session and the USER menu item for a display session. To generate one additional report, you replace module ERBGUS99 with your data gatherer and ERBRUS99 with your data reporter. Specifying USER then causes your own report to be generated. Should you want to obtain more than one user report, you must add an entry to the option list or menu list as well as supply a data-gathering and a data-reporting routine. Data gathered for your routine can be reported either during the session or during execution of the Postprocessor.

During a Monitor II TSO/E display session, with TSO/E installed, a user exit enables your installation to verify that a terminal user is authorized to use RMF. See “TSO terminal user authorization” on page 155 for an explanation of this user exit.

Monitor I session user reports

Programming Interface information

To gather and report data relevant to your installation during a Monitor I session, RMF provides both the EXITS option and installation exits at various points during Monitor I session processing. When EXITS is specified, you can:
- Initialize for the other user exit routines
- Sample fixed CSA, SQA, or nucleus data at each RMF cycle
- Perform interval processing, for example, reduce sampled data, examine system state indicators, format SMF records to be written to the SMF data set or passed to your report writer
- Write reports during a session
- Handle termination processing for the other installation exits
- Write reports during execution of the Postprocessor.

In addition, you can use the Monitor I session tracing routines to trace the contents of a fixed SQA, CSA, or nucleus field regardless of whether or not EXITS is specified.

Guidelines

Each of the user functions is described in detail in the following sections. The following guidelines apply to Monitor I user exit routines:

- All of the user exit routines must be reenterable.
- All user-written exit routines receive control in 31-bit addressing mode.
- The routines must save registers when they receive control and restore registers when they return control. Register 13 contains the address of the register save area; register 14 contains the return address; and register 15 contains the entry address.
- One input parameter that RMF passes to each user exit routine (except the tracing routine and the Postprocessor user exit) is the address of a two-word area reserved for the use of your routines. Because these words provide a means of communication between your exit routines, their use should be controlled by conventions agreed upon by your installation.
- RMF passes a phase parameter to each user exit routine except the sampler, the tracing routine, and the Postprocessor user exit. This phase parameter indicates which RMF phase is invoking the user exit.

RMF provides dummy routines for all Monitor I session exits that are not used.

Caution

Because all of the user exit routines except ERBMFRUR (the report writer) run in supervisor state with a key of 0, your installation must carefully control their use. Program errors that cause an exit routine to overlay system areas could bring down the system.

Initialization for Monitor I session user exit routines

The initialization user exit is ERBMFIUC. It is called at the start of a Monitor I session and whenever the Monitor I session options are modified. Use this exit to perform any initialization the other installation exits require, such as building a control block structure.

When the exit routine gets control, register 1 points to a three-word address list. The first address points to the two-word area reserved for use by your routines. This same two-word area is passed to all the user exit routines and can be used for communication between them. The second address points to the RMF phase parameter, a full-word field that is always X‘4:’, indicating that the exit is called during Monitor I session initialization. The third address points to a word that is relevant only when you are providing a routine to sample data at each cycle; one of the functions your initialization routine will perform is to put the address of the user sampler in this word. Figure 15 illustrates the input parameter structure.

When the initialization routine is entered, the system is in supervisor state, and all interrupts are enabled. ERBMFIUC runs in key 0.

Special initialization procedures are required when your user routines include a sampling routine to sample data at each cycle; see “Sampling Data at Each Cycle.” When you have a user sampler, your initialization routine **must** do the following:

- The user sampling routine must be loaded and page fixed. You must use the PGSER macro to page fix the user sampler routine because the sampler code runs disabled.
- The address of the user sampling routine must be placed in the third input parameter.
- All storage the sampler routine will require must be obtained; this storage must be obtained from SQA (subpool 245).
- The address of the SQA storage obtained must be placed in one of the two user words. The choice depends on the conventions established at your installation.

When you have completed the initialization required by all the installation exits, return control by branching on register 14.

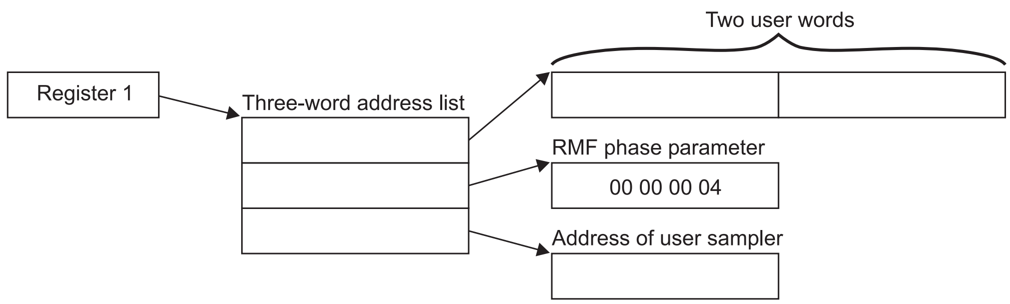


Figure 15. ERBMFIUC Input Parameter Structure

Sampling data at each cycle

To sample data at each cycle, the steps described earlier for initialization must be performed to load and page fix the user sampler routine. A user sampler routine is activated at each cycle only when another measurement that includes a sampling routine is activated. These measurements include paging activity, page/swap data set activity, channel path activity, I/O queuing activity, device activity, and trace activity. At least one of these measurements must be specified to enable RMF to invoke your user sampler.

When the sampler gets control, register 1 points to a two-word area. One of these words, selected by your installation, contains the address of the storage area obtained for the sampler by ERBMFIUC. Figure 16 illustrates the input parameter structure.

When the user sampler is entered, the system is in supervisor state, and all interrupts are disabled. The routine runs in key 0. It can sample any fixed data in CSA, SQA, or the nucleus; no other data areas can be sampled. You place the data sampled in the storage area obtained by ERBMFIUC and passed to you when your routine is invoked. This storage area is always in SQA (subpool 245). At the end of the RMF interval, RMF passes the address of the storage area to the user interval processing routine. Should your routine cause a page fault, the Monitor I session terminates abnormally with an abend code of 0FE.

When your sampling is completed, return control by branching on register 14.

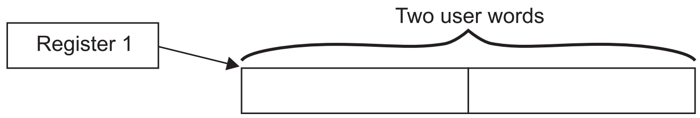


Figure 16. User Sampler Input Parameter Structure

Note: The user sampler must reside in SYS1.SERBLPA. See “Adding Your Routines to RMF” later in this chapter.

Interval processing

The interval processing user exit is ERBMFDUC. It is invoked at the start of the Monitor I session and at the end of each RMF interval.

When the exit gets control, register 1 points to a two-word address list. The first address points to the two-word area reserved for use by your routines. When these routines include a user sampler, one of these words, selected by your installation, will contain the address of the sampled data. The second address points to the RMF phase parameter. This parameter is a full word that contains X‘4’ when the exit is called during Monitor I session initialization, X‘8’ when the exit is called at the end of an RMF interval, or X‘C’ when the exit is called at the end of an RMF interval for which data collection was skipped. Figure 17 illustrates the input parameter structure.

When the interval processing exit routine is entered, the system is in supervisor state, and all interrupts are enabled. The routine runs in key 0. The routine can process the data generated by the user sampler. It can also collect its own data from system control blocks or system state indicators and format an SMF record. The SMF record can be written to the SMF data set; see *z/OS MVS System Management Facilities (SMF)* for details on using the SMFEWTM macro instruction to write a user SMF record.

The SMF record or a record your routine formats as agreed by convention between ERBMFDUC and ERBMFRUR (the report writer exit routine) can be printed by your report writer. Your routine can format SMF record output, report record output, or both. When your routine formats any records to be printed by your report writer, the address of the formatted records must be placed in the user word selected by your installation. Because the user words are passed to your report writer, the records can then be printed in a formatted report.

When the length of the RMF interval exceeds 99 minutes, which can occur when RMF is not dispatched at the end of an interval, data collection for the interval is skipped. Because there is no data collected, RMF does not call the report writer user exit (ERBMFRUR); instead, ERBMFDUC is called twice. The phase parameter is X‘8’ for the first invocation of the exit routine and X‘C’ for the second. When the exit routine is called with a phase parameter of X‘C’, your routine must free the storage areas normally freed by ERBMFRUR. RMF issues a message to notify the operator that data collection was skipped for the interval.

When your routine has completed processing, return control by branching on register 14.

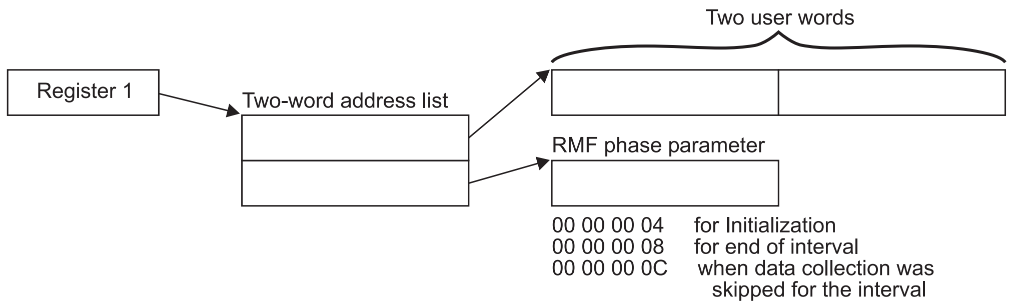


Figure 17. ERBMFDUC Input Parameter Structure

Report writing during session processing

The report writer exit is ERBMFRUR. It is called once during the Monitor I session report writing phase.

When the exit gets control, register 1 points to a two-word address list. The first address points to the two-word area reserved for use by user routines. The second address points to the RMF phase parameter, which is always X'10' for the report writer. Figure 18 illustrates the input parameter structure.

When the report writer exit is entered, the system is in problem state, and all interrupts are enabled. The routine runs in the user key 8. The user word your installation selects contains the address of the formatted records built by ERBMFDUC. Because all of your installation's exit routines use these words, the report writer must not alter their contents. Report writer processing must obtain output space for the printed reports, then write the reports for subsequent printing. Before terminating, the routine must free the storage that contained the records formatted by ERBMFDUC.

When the report writer completes its processing, return control by branching on register 14.

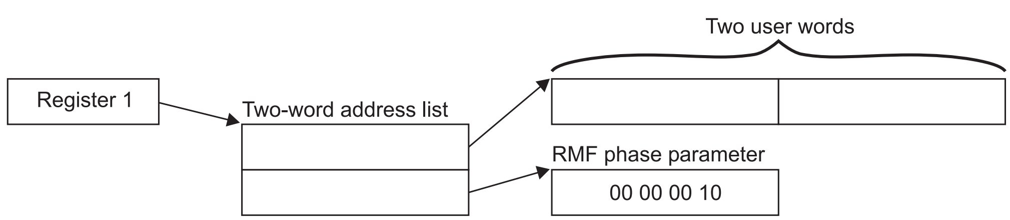


Figure 18. ERBMFRUR Input Parameter Structure

Termination

The termination exit is ERBMFTUR. It is called when the Monitor I session is terminated.

When the exit gets control, register 1 points to a two-word address list. The first address points to a two-word area reserved for use by your routines. The second address points to the RMF phase parameter, which is always X'C' for termination. Figure 19 illustrates the input parameter structure.

When the termination routine is entered, the system is in supervisor state, and all interrupts are enabled. The routine runs in key 0. You would use this exit to page free any user samples or data areas and to free any user SQA data areas obtained by the other exits, with one exception: during termination processing, ERBMFTUR gets control before the report writer exit (ERBMFRUR). Therefore, it must free only the SQA and global storage the other user routines obtained, but it **must** not free the storage the interval processing routine (ERBMFDUC) used to build records to be passed to the report writer. The address of this storage will be in the user word selected by your installation.

When the termination routine has completed processing, return control by branching on register 14.

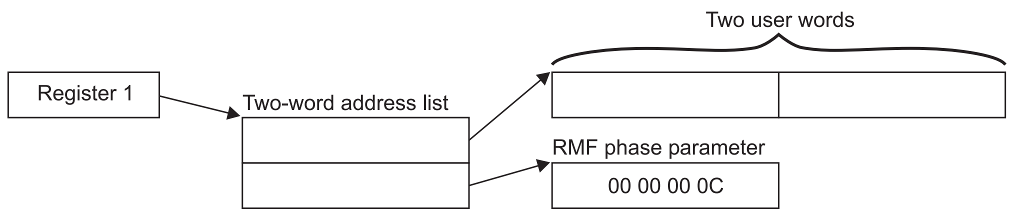


Figure 19. ERBMFTUR Input Parameter Structure

Tracing your own field

Whenever the TRACE option is in effect during a Monitor I session, you can use the tracing routines to trace the contents of any SQA, CSA, or nucleus field that you require. The EXITS option, required to generate your own reports, is not required to use the trace facilities because the trace exit, ERBTRACE, is called whenever the TRACE option is in effect.

The field that you want to trace must be fixed in CSA, SQA, or the nucleus; it cannot contain negative values; and it must be from one to four bytes in length. Once you have selected your field, there are two steps required to enable RMF to trace the contents of the field. After you have performed these steps, you can then specify the name in the field name portion of the TRACE option. The steps can be performed in any order, but both must be done before you can use RMF to trace the field.

Step 1 -- Defining the name to RMF

To define the name, you must add four fields to the RMF CSECT ERBMFTTB, which contains the names RMF recognizes as valid for tracing. The fields you must add are:

1. The name of the field to be traced. The name can be from one to eight bytes long. It must not be the same as any name already recognized by RMF. When the name of the field is less than eight bytes long, it must be padded on the right with blanks to a length of eight bytes.
2. The length of the name. This field is one byte long; the value must be from 01 to 08.
3. A one-byte constant that always contains the value X'DC'.
4. The length of the field to be traced. This field is one byte long; the value must be from 01 to 04.

One byte of binary zeroes must follow the last entry to be added; the byte of binary zeroes indicates the end of the variable-length trace table. Figure 20 shows an example of how to superzap ERBMFTTB to add a new name for tracing. The parenthesized numbers in the text refer to the parenthesized numbers in the figure. The example adds a nucleus field named MYDATA (1) that is two bytes in length (4) to the list of names valid for RMF tracing. The name is six bytes long (2), and the required constant is also supplied (3). A byte of binary zeroes (5) indicates the end of the trace table. Adding the name definition to ERBMFTTB causes RMF to pass the name to ERBTRACE during each tracing cycle. The four fields must be added for each name you want RMF to trace; only the last entry must be followed by the byte of binary zeroes.

```
//ZAP       JOB         MSGLEVEL=1
//SS        EXEC        PGM=AMASPZAP
//SYSPRINT  DD          SYSOUT=A
//SYSLIB    DD          DSN=SYS1.SERBLINK,DISP=SHR
//SYSIN     DD          *
    NAME    ERBMFMFC        ERBMFTTB
    VER     040C            0040D7C1
    REP     040C            D4E8C4C1E3C14040 (1)
    REP     0414        06  (2)
    REP     0415        DC  (3)
    REP     0416        02  (4)
    REP     0417        00  (5)
/*
```

Figure 20. Example of Adding a Name to ERBMFTTB

Step 2 -- Replacing ERBTRACE

The tracing user exit is ERBTRACE. The function of ERBTRACE is to return to RMF the address of a valid user field. It is called by the RMF tracing routine whenever it encounters a trace name that is not the name of a field in the SRM domain table. To trace your own field, you must replace ERBTRACE with your own routine and link edit your ERBTRACE with the RMF CSECT ERBMFITR.

When ERBTRACE gets control, register 1 points to a two-word address list. The first address points to an eight-byte field that holds the name to be validated. The second address points to a full word to be used by ERBTRACE to return the address of the user field to RMF. Figure 21 illustrates the input parameter structure.

When ERBTRACE is entered, the system is in supervisor state, and all interrupts are enabled. The routine runs in key 0. It must examine the field name passed to it by RMF to determine if the name is a user field name. When the name is a valid user name, place the address of the field to be traced in the first parameter, set a return code of zero in register 15, and return control. If the name is not one recognized as a valid user name, always set a non-zero return code in register 15 before returning control. The non-zero return code tells RMF to process the name.

When your processing is completed, return control by branching on register 14.

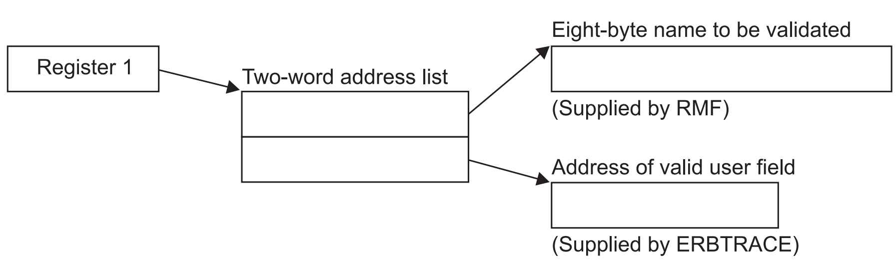


Figure 21. ERBTRACE Input Parameter Structure

Report writing by the Postprocessor

The Postprocessor user exit is ERBMFPUS. It is called during post-processing at the point when the Postprocessor first encounters each SMF record. When the exit gets control, register 1 points to a three-word address list:

- The first address points to a full word that contains the address of the SMF record to be processed.
- The second address points to a full word reserved for the use of your routine. The user word contains zeros the first time the exit is called, and the Postprocessor does not modify its contents between invocations of the user exit routine. Thus, the word can be used to save information, such as the address of a DCB, that is needed by a subsequent invocation of the user exit routine.
- The third address points to a FIXED(8) field, which contains X'01' for EOF.

Figure 22 illustrates the input parameter structure.

When the Postprocessor user exit is entered, the system is in problem state and all interrupts are enabled. The routine runs in the user key 8.

Your routine examines the SMF record passed to you, performs any required processing, and set a return code in register 15. The return code depends on the action you want the Postprocessor to take. A return code of 0 tells the Postprocessor to continue processing the SMF record. A return code of 4 tells the Postprocessor to ignore the SMF record; set a return code of 4 when the exit routine has, for example, processed the record or determined that it should not be processed. A return code of 8 indicates that the Postprocessor should terminate.

The processing your exit performs can consist of formatting the data in the records that the interval processing user exit routine (ERBMFDUC) generates into a printed report. Your exit could also screen the SMF records that the Postprocessor encounters to determine which records are to be included in any reports generated by the Postprocessor, or it could use the SMF records RMF generates as input to your own report. Because all SMF records are passed to the user exit, ERBMFPUS could also be used to incorporate any SMF data reduction routines used at your installation into the RMF Postprocessor function.

When your routine has finished processing, set the appropriate return code in register 15 and return control to the RMF Postprocessor by branching on register 14.

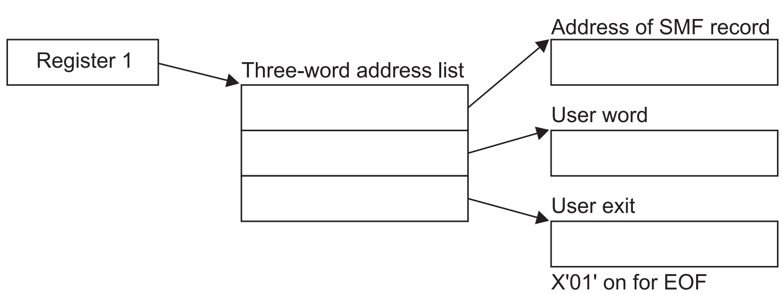


Figure 22. ERBMFPUS Input Parameter Structure

Adding your routines to RMF

Before your Monitor I session user exit routines can be tested and used, they must be assembled and link edited with the appropriate RMF modules. If you are using your private libraries, you have to ensure that they are concatenated in front of the distributed RMF libraries. Figure 23 shows sample JCL for performing the required link edit for all user routines except the sampler routine. If you have a user sampler, a separate link edit is required; a sample is shown in Figure 24.

```
//LINKEXIT JOB   MSGLEVEL=1
//LINK0001 EXEC PGM=IEWL,PARM='MAP,XREF,REUS,RENT,REFR,NCAL'
//SYSPRINT DD    SYSOUT=A
//SYSLMOD  DD    DSN=SYS1.SERBLINK,DISP=(OLD,KEEP)
//SYSUT1   DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(TRK,(20,5))
//SYSLIN   DD    *
    (ERBMFIUC object deck)
    ENTRY   ERBMFIUC
    NAME    ERBMFIUC(R)
    (ERBMFDUC object deck)
    ENTRY   ERBMFDUC
    NAME    ERBMFDUC(R)
    (ERBMFRUR object deck)
    ENTRY   ERBMFRUR
    NAME    ERBMFRUR(R)
    (ERBMFTUR object deck)
    ENTRY   ERBMFTUR
    NAME    ERBMFTUR(R)
    (ERBTRACE object deck)
    INCLUDE SYSLMOD(ERBMFITR)
    ENTRY   ERBMFITR
    NAME    ERBMFITR(R)
    (ERBMFPUS object deck)
    ENTRY   ERBMFPUS
    NAME    ERBMFPUS(R)
/*
```

Figure 23. Replacing Installation Exits

```
//LINKEXIT JOB   MSGLEVEL=1
//LINK0001 EXEC PGM=IEWL,PARM='MAP,XREF,REUS,RENT,REFR,NCAL'
//SYSPRINT DD    SYSOUT=A
//SYSLMOD  DD    DSN=SYS1.SERBLPA,DISP=(OLD,KEEP)
//SYSUT1   DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(TRK,(20,5))
//SYSLIN         DD    *
    (user sampler object deck)
    ENTRY   entry name
    NAME    sampler name
/*
```

Figure 24. Adding a User Sampler

End of Programming Interface information

Monitor II session user reports

Programming Interface information

RMF generates a Monitor II session report by invoking a data-gathering module and a data-reporting module in response to either:
- a menu item identifying a display session report
- an option identifying a background session report

From an external viewpoint, the menu item and the option are different because they are used during different types of sessions, have slightly different syntax, and produce either display output or printed output. However, from an internal point of view, the menu item and the option are very similar. The valid menu items for a display session are listed in the RMF CSECT ERBFMENU.

Note: If you are running the Kanji version of RMF, the corresponding CSECT is ERBJMENU, and you should ensure that both CSECTs stay synchronized.

The options for a background session are listed in the RMF CSECT ERBBMENU. The formats of the entries in each list are identical. When an option or menu item is specified during a session, RMF uses the data entry for the report in the list appropriate for the session type to verify that the option or menu item is valid and to load the required data gatherer and data reporter modules.

Each list contains an entry called USER that enables you to add a single user report. When USER is specified, RMF loads modules ERBGUS99, the data gatherer for USER, and ERBRUS99, the data reporter for USER. By replacing these two modules with your own routines, you can add a single report to the Monitor II reports provided by RMF. This process is described later in this chapter under "Coding a User Report."

The data gathering module and the data reporting module communicate through a type 79 SMF record. The data gatherer formats the record and completes the required data fields. The data reporter uses the data in the record to generate a formatted report for printing or display. See "SMF Record Type 79."

To add more than one Monitor II session report, you must, in addition to providing a data gatherer and a data reporter, add an entry to ERBFMENU for a display session report and to ERBBMENU for a background session report. Then, when your option or menu item is specified during a session, RMF will load your data gatherer and data reporter to generate the report. The process to follow to add an entry to the option list and menu list is described later in this chapter under "Installing a User Report."

Guidelines

Each of the user functions is described in detail in the following sections. The following guidelines apply to all Monitor II user exit routines.

- All of the user exit routines must be reenterable.
- All user-written exit routines receive control in 31-bit addressing mode.
- The routines must save registers when they receive control and restore registers when they return control. Register 13 contains the address of the register save area; register 14 contains the return address; and register 15 contains the entry address.
- All of the user exit routines receive control in problem state, key 8.

SMF record type 79

SMF record type 79 must be used to record data gathered by a user data gathering routine. Figure 25 shows the layout of the record sections that are common to all Monitor II data gatherers, whether coded by a user or provided by RMF. The figure illustrates the layout of these common sections by showing the expansion of the RMF mapping macro ERBSMF79.

The fields in the common sections fall into three categories. Each category is indicated by a letter in the figure that corresponds to the letters in the following text:

A The fields that the RMF routines fill in before the data gathering routine is invoked.

B The fields that the data gathering routine must fill in during its processing. (See “Relocate Blocks” later in this section.)

C The fields that the RMF routines will fill in when the RECORD option is in effect. RMF completes these fields after the data gatherer returns control but before the record is written to the SMF data set. During a display session or a background session when NORECORD is in effect, these fields are not completed because the record is not actually written to the SMF data set.

Before invoking the data gatherer, RMF calculates the length of the storage buffer required for the record, as described later under “Relocate Blocks,” obtains a buffer for the record, and fills in some of the common section fields. The address of the SMF record buffer is passed to the data gatherer. The data gatherer fills in some fields in the common section and all of the data section of the record.

```
    ********************  COMMON SMF HEADER  *****************************
        SMF79HDR   DSECT
    C   SMF79LEN   DS      BL2      RECORD LENGTH
        SMF79SEG   DS      BL2      SEGMENT DESCRIPTOR
    C   SMF79FLG   DS      BL1      HEADER FLAG BYTE
        SMF79RRF   EQU     X'80'    NEW SMF RECORD FORMAT IF=1
        SMF79SUT   EQU     X'40'    SUBTYPE UTILIZED IF=1
        SMF79ESA   EQU     X'08'    MVS/ESA IF=1
        SMF79VXA   EQU     X'04'    MVS/XA IF=1
        SMF79OS    EQU     X'02'    OPERATING SYSTEM IS OS/VS2
        SMF79BFY   EQU     X'01'    SYSTEM IS RUNNING IN PR/SM MODE
        SMF79PTN   DS      BL1      PR/SM PARTITION NUMBER
    C   SMF79RTY   DS      BL1      RECORD TYPE
        SMF79TME   DS      BL4      TOD RECORD WRITTEN
        SMF79DTE   DS      PL4      DATE RECORD WRITTEN
    C   SMF79SID   DS      CL4      SYSTEM ID FROM INSTALLATION
    C   SMF79SSI   DS      CL4      SUBSYSTEM ID (RMF)
    B   SMF79STY   DS      BL2      SUBTYPE
    A   SMF79TRN   DS      BL2      NUMBER OF TRIPLETS IN THIS RECORD
                   DS      BL2      RESERVED
    A   SMF79PRS   DS      BL4      OFFSET TO RMF PRODUCT SECTION
    A   SMF79PRL   DS      BL2      LENGTH OF RMF PRODUCT SECTION
    A   SMF79PRN   DS      BL2      NUMBER OF RMF PRODUCT SECTIONS
      *********************  INDIVIDUAL HEADER EXTENSION *******************
    A   SMF79MCS   DS      F -      OFFSET TO MONITOR II CONTROL SECTION
    A   SMF79MCL   DS      H -      LENGTH OF MONITOR II CONTROL SECTION
    A   SMF79MCN   DS      H -      NUMBER OF MONITOR II CONTROL SECTION
    B   SMF79ASS   DS      F -      OFFSET TO DATA SECTION
    B   SMF79ASL   DS      H -      LENGTH OF DATA SECTION
  B A   SMF79ASN   DS      H -      NUMBER OF DATA SECTION
    A   SMF79DCS   DS      F -      OFFSET TO DATA CONTROL SECTION
    B   SMF79DCL   DS      H -      LENGTH OF DATA CONTROL SECTION
    B   SMF79DCN   DS      H -      NUMBER OF DATA CONTROL SECTION
        SMF79QSS   DS      F -      OFFSET IOQ GLOBAL SECTION
        SMF79QSL   DS      H -      LENGTH IOQ GLOBAL SECTION
        SMF79QSN   DS      H -      NUMBER IOQ GLOBAL SECTION
```

Figure 25. ERBSMF79 Mapping Macro Expansion (Part 1 of 2)

```
   *********************  COMMON SMF PRODUCT SECTION ********************
    SMF79PRO   DSECT
C   SMF79MFV   DS      CL2      RMF VERSION NUMBER, WITH
    *                                    INTRODUCTION OF THE MVS
    *                                    SOFTWARE LEVEL, THE FORMAT
    *                                    CHANGES TO PACKED (VRLF),
C   SMF79PRD   DS      CL8      PRODUCT NAME
    SMF79IST   DS      PL4      TOD MONITOR 1 INTERVAL START: 0HHMMSSF
C   SMF79DAT   DS      PL4      DATE MONITOR 1 INTERVAL START: 00YYDDDF
    SMF79INT   DS      PL4      DURATION OF MONITOR 1 INTERVAL: MMSSTTTF
    *          DS      BL2      RESERVED
B   SMF79SAM   DS      BL4      NUMBER OF SAMPLES
    *          DS      BL2      RESERVED
    SMF79FLA   DS      BL2      FLAGS
    SMF79ISS   DS      X'40'    INVALID SAMPLES TO BE SKIPPED
    SMF79M3R   DS      X'20'    RECORD WAS WRITTEN BY RMF MONITOR III
    SMF79ISM   DS      X'10'    INTERVAL WAS UNDER SMF CONTROL
    *          DS      BL4      RESERVED
B   SMF79CYC   DS      PL4      CYCLE IN PACKED DECIMAL 000TTTTF
B   SMF79MVS   DS      CL8      MVS SOFTWARE LEVEL
B   SMF79IML   DS      BL1      TYPE OF PROCESSOR COMPLEX ON WHICH DATA IS MEASURED
B   SMF79PRF   DS      XL1      PROCESSOR FLAGS
B   SMF79QES   EQU     X'80'    EQUIPPED WITH EXPANDED STORAGE
B   SMF79CNE   EQU     X'40'    EQUIPPED WITH ESCON CHANNEL
B   SMF79DRC   EQU     X'20'    ESCON DIRECTOR IN CONFIG.
B   SMF79EME   EQU     X'10'    SYSTEM IS RUNNING IN Z/ARCHITECTURE
B   SMF79PTN   DS      BLI      PR/SM PARTITION NUMBER
    SMF79SLR   DS      BL1      SMF RECORD LEVEL
    SMF79IET   DS      CL8      INTERVAL EXPIRATION TIME TOKEN
   *********************  MONITOR II CONTROL SECTION  *******************
    R79CHL     DSECT            COMMON RECORD 79 HEADER
B   R79GTOD    DS      XL4 -    DATA GATHERER CALL TOD
B   R79LF2     DS      XL1 -    FLAG BYTE
    R79PAR     EQU     X'80'    NOT ENOUGH RELOCATE SECTION TO
    *                           COMPLETE DATA GATHERING
    R79SG      EQU     X'40'    REPORT TO BE SORTED BY SG
    R79RV1     DS      XL1 -    RESERVED
C   R79SES     DS      CL2 -    SESSION NAME
    R79RSV     DS      XL2 -    RESERVED
    R79USER    DS      XL2 -    USER FIELD
C   R79RID     DS      CL8 -    MEASUREMENT NAME
C   R79CTXTL   DS      XL2 -    LEN OF COMMAND TEXT
C   R79CTEXT   DS      CL32 -   COMMAND TEXT
C   R79DTXTL   DS      XL2 -    LEN OF DEFAULT DR TEXT
C   R79DTEXT   DS      CL32 -   DEFAULT DR TEXT
C   R79IST     DS      CL4 -    MON III INTERVAL START TIME :0HH MMSSF
   ********************* DATA  SECTION  ********************************
    R799LCU    DS      BL2      LOGICAL CONTROL UNIT NUMBER 0 TO 255
    R799SGN    DS      CL8      STORAGE GROUP NAME
```

Figure 25. ERBSMF79 Mapping Macro Expansion (Part 2 of 2)

Relocate blocks

The data section of SMF record type 79 is unique to each report. It is composed of one or more data sections called **relocate** blocks and, possibly, one data control section. A relocate block is the portion of the SMF record that contains the data for one report data line. A record for a row report has one relocate block. A record for a table report has multiple relocate blocks; for example, the SMF record for the address space state data report includes one relocate block for each address space included in the report. When your SMF record has multiple relocate blocks and you are gathering data that applies to all of them, you can, instead of reporting the data in each relocate block, place this common data in a data control section, as described later under “Data Control Section”.

The format of the data in the relocate block depends on the report you are generating. You set the format that best meets your needs. When you are generating a table report, the SMF record consists of multiple relocate blocks, and each relocate block must have the same length.

When you add a menu item to ERBFMENU or an option to ERBBMENU, the entry that describes the new report must include a field that specifies the length of the relocate block, the maximum number of possible relocate blocks, and the length of the data control section. For information on how to add an entry to ERBFMENU or ERBBMENU, see "Using the PICTURE macro" on page 153. To determine the storage to allocate, RMF multiplies the length of the relocate block by the maximum number of relocate blocks and adds this value to the length of the data control section and the common section. The result of this computation is the maximum possible length of the SMF record, and RMF allocates a buffer for the record that is equal in size to the maximum length.

To determine the actual length of the SMF record, the data gatherer must complete the fields in the individual header extension section that describe the offset, length, and number of data sections and the data control sections. After the data gatherer has completed its processing and returned control, RMF uses these values to determine the length of the SMF record to be written to the SMF data set, a calculation that is performed only when the RECORD option is in effect for a background session. Note that the value your routine sets in SMF79ASL and the value specified for RBLEN in the PICTURE macro for the report should be identical.

Other fields in the common section that the data gather completes are R79GTOD and SMF79STY. R79GTOD must contain a packed decimal value that indicates the time when the data gatherer was invoked, in the form 0hhmmssF, where F is the sign. SMF79STY can contain the subtype number of the SMF record that you are creating. You use this number as a unique identifier for each record subtype that you create; no subtype number should be less than 1000.

The maximum length of an SMF record is 32,756 bytes; any records that exceed this length are truncated before they are written to the SMF data set. Truncation, which can occur only during a background session when the RECORD option is in effect, occurs at the last relocate block boundary within the maximum length. When truncation occurs, RMF adjusts the field indicating the capacity of the buffer (SMF79ASN) to indicate the actual number of relocate blocks in the record. If no truncation occurs, RMF leaves SMF79ASN unchanged.

Data control section

A data control section is useful when your SMF record might have many relocate blocks and some of the data you are gathering is common to all of them. For example, the channel path Monitor II control section (subtype C) uses a control section to record the number of times the channel was sampled. To use a data control section:

1. Set the value for the FBLEN parameters on the PICTURE macro instruction for your report, as described under "Using the PICTURE macro" on page 153.
2. Format the data control section to hold the common data.
3. Place it between the Monitor II control section and the data section. SMF79DCS contains the offset at which it should start.
4. Set SMF79DCL and SMF79DCN to the length and number of the data control sections.

5. Set the offset to the first data section SMF79ASS to point to the end of the data control section.

When a data control section is *not* used:
1. Set SMF79ASS to the value in SMF79DCS.
2. Set SMF79DCL and SMF79DCN to 0.

Coding a user report

To add a Monitor II report, you must code your own data gatherer module and data reporter module. These modules can reside in SYS1.SERBLINK, SYS1.SERBLPA, a steplib, a joblib, a tasklib, or a library in a linklist.

The primary means of communicating data between the gatherer and the reporter is the type 79 SMF record. The gatherer collects data from whatever areas it can access (it runs in problem state with a key of 8) and places the data in the SMF record. The reporter takes the data from the SMF record, formats it for output, and passes it to the RMF putline routine. During a Monitor II background session, the data reporter would be called when the REPORT option is in effect. When NOREPORT and RECORD are in effect, RMF writes out the SMF records that the data gatherer formats, and the data reporter is not invoked. Your data reporter can be invoked at a later time by the Postprocessor.

A Monitor II session report can have operands that the report user specifies when requesting the report. Any operands specified when a report is requested are passed to both the data gatherer and the data reporter. The defaults established for each possible operand are specified in the option list or menu list entry for the report; these defaults are also passed to both the data gatherer and the data reporter. Your routines can also include hard-coded default operands.

Because the option list and menu item list are in different RMF control sections, you can set different default operands for a background session and a display session. Each list entry contains separate fields for the data gatherer default operands and the data reporter default operands; you can thus set different default operands for the data gatherer and the data reporter. For example, the default operands for the RMF address space state data gatherer module cause data to be gathered on all address spaces in the system; to limit the actual output produced, the defaults for the reporter cause only the active address spaces to be reported. "Using the PICTURE Macro" describes how to specify default operands.

RMF passes parameters to both the gatherer and reporter; these parameters include a subpool number that indicates the subpool from which the routines should obtain the storage they require, and two user words that can be used for communication between the data gatherer and the data reporter. Because the same two words are passed to both routines, use of these words must be governed by conventions established by your installation.

Note: A system status line precedes each display report supplied by IBM. RMF obtains the data for this line before it invokes the data gatherer for the report. RMF will generate the same system status line before each user-coded display report.

Data gatherer

The data gatherer runs in problem state, with a key of 8, and in 31-bit addressing mode. The data gatherer must be reenterable. It receives control by a BALR instruction and must save the registers when it receives control and restore the registers when it returns control. Register 13 contains the address of the register save area; register 14 contains the return address; and register 15 contains the entry address.

Upon entry to the data gatherer, register 1 points to a contiguous list of seven addresses that point to seven input parameters. The first address points to the first parameter, the second address points to the second parameter, and so forth. The input parameters are:

First Parameter: A fullword entry code that must always be X‘2’.

Second Parameter: The operands, if any, specified by the report user when he requested the report, in the form:

LL	text

LL A two-byte length field indicating the length of the following text (does not include the two bytes of LL)

text A character string of up to 32 characters containing the input operands

When the report has no operands or the report request did not include operands, LL is set to zeros.

RMF determines the operands to be placed in **text** by scanning the report request. The first non-blank character after the report name is assumed to be the first character of the operand field. The next blank character is assumed to mark the end of the operand field.

Third Parameter: The default operands from ERBFMENU or ERBBMENU, in the form:

LL	text

LL A two-byte length field indicating the length of the following text (does not include the two bytes of LL)

text A character string of up to 32 characters containing the default operands

When the report has no operands or no default operands, LL is set to zeros.

Fourth Parameter: The pointer to the SMF record buffer where your routine is to place the data it gathers.

Fifth Parameter: The first of the two words reserved for the use of your routines.

Sixth Parameter: The second of the two words reserved for the use of your routines.

Seventh Parameter: A byte containing the number of the subpool to use when you issue a GETMAIN to obtain the storage your routine requires.

The processing your data gathering routine performs is determined largely by the nature of the report for which you are gathering data. This processing should include a validation of the entry code in the first parameter to verify that it is X‘2’. If it is not, set a return code of 8 in register 15 and return control.

If the report has operands that can be specified when the report is requested, check the second input parameter to determine if the request specified operands. If it did, validate the syntax of the operands; if the syntax is invalid, set a return code of 4 in register 15 and return control. If the request did not specify operands, verify the syntax of the default menu operands passed as the third input parameter; if the syntax is invalid, set a return code of 24 in register 15 and return control.

Your routine should complete the required fields in the SMF record common section (the **B** fields in Figure 25), using the RMF mapping macro ERBSMF79 to access the fields in the common section. The address of the storage buffer obtained for your record is passed in the fourth input parameter. Your routine would gather the data required and format the data section of the record as agreed upon by convention between the data gatherer and the data reporter. Should your routine locate no data that is applicable to the report requested, set a return code of 16 in register 15 and return control.

When your routine has finished processing, set a return code in register 15 and return to the caller by branching on the contents of register 14. Table 14 shows the possible return codes, their meaning, and the action RMF takes in response. These return codes apply to both the data gatherer and the data reporter.

Note: If your report will be run only during a display session, you can perform both the data gathering function and the data reporting function in the data reporter module. In this case, your data gatherer's only function would be to set a return code of zero in register 15. However, if you choose to perform both functions in the data reporter module, your report cannot run during a Monitor II background session and, during a display session, you will not be able to use the recall command to re-display your report.

Table 14. Return Codes from the Data Gatherer and Data Reporter

Code	Meaning	RMF Response (Display Session)	RMF Response (Background Session)
0	Successful completion.	The session continues.	The session continues.
4	Invalid operand syntax.	The command is displayed as entered.	Message ERB409I is issued. The current measurement continues if the error was detected by the data reporter and RECORD is in effect; otherwise, the measurement is discontinued. The session continues. The operator can modify the session options.
8	Invalid entry code.	Abend - the user code is 1402.	Abend - the user code is 1402.
12	I/O error.	Messages ERB403I and ERB404I are displayed, including the SYNAD text.	The current measurement continues when RECORD is in effect, but no subsequent reports are printed; otherwise, the measurement is discontinued. The session continues.
16	No data found.	Message ERB405I is displayed.	Message ERB405I is issued. No report or SMF record is produced for this interval. All measurements continue.

Table 14. Return Codes from the Data Gatherer and Data Reporter (continued)

Code	Meaning	RMF Response (Display Session)	RMF Response (Background Session)
20	ESTAE macro failed.	Message ERB406I is displayed.	Message ERB406I is issued. The current measurement continues if the error was detected by the data reporter and RECORD is in effect; otherwise, the measurement is discontinued. The session continues.
24	Menu default operand syntax error.	Message ERB407I is displayed, including the menu defaults and advice to retry the report, specifying all operands.	Message ERB407I is issued. The current measurement continues if the error was detected by the data reporter and RECORD is in effect; otherwise, the measurement is discontinued. The session continues.
28	The amount of data to be gathered exceeds the number of available relocate blocks.	Message ERB411I is displayed.	Message ERB411I is issued. The report or SMF record produced for the interval includes only the data gathered before the condition was detected. All measurements continue.
32	Monitor I report not active.	Message ERB412I is displayed.	Message ERB412I is issued. No report or SMF record is produced for the interval. All measurements continue.
36	Monitor I interval is less than Monitor II interval.	Message ERB413I is displayed.	Message ERB413I is issued. No report or SMF record is produced or the interval. All measurements continue.
40	The SRM's store channel path status facility is not active. Used by channel path activity (CHANNEL) report.	Message ERB264I is displayed.	Message ERB264I is issued. No report or SMF record for channel path activity is produced; the current measurement is discontinued. All other measurements continue.
44	Report option no longer applicable.	Message ERB434I is displayed.	Message ERB434I is issued. No SMF record is produced for this report. All other measurements continue.
48	No transaction data available.	Message ERB435I is displayed.	Message ERB435I is issued. No SMF record is produced for this report. All other measurements continue.
52	SRM mode changed - interval skipped.	Message ERB436I is displayed.	Message ERB436I is issued. No SMF record is produced for this report. All other measurements continue.
>56	Unexpected.	Message ERB408I is displayed.	Message ERB408I is issued. The current measurement continues if the error was detected by the data reporter and RECORD is in effect; otherwise, the measurement is discontinued. The session continues.

Data reporter

The data reporter runs in problem state, with a key of 8, and in 31-bit addressing mode. The data reporter must be reenterable. It receives control by a BALR instruction and must save the registers when it receives control and restore the registers when it returns control. Register 13 contains the address of the register save area; register 14 contains the return address; and register 15 contains the entry address.

The data reporter formats each line in the report, using the data placed in the type 79 SMF record by the data gatherer. The RMF putline routine is used to perform the actual output operation.

Because the putline routine handles the actual output operations, your data reporter can function identically during a background session, a display session, a display session in hardcopy mode, or an execution of the Postprocessor. The putline routine writes the line to a logical screen buffer for a display session, to a logical screen buffer and an output data set for a display session in hardcopy mode, or to an output data set for a background session or an execution of the past processor. For a display session, the screen is updated to show the lines collected by the putline routine when your data reporter returns control. Note that RMF handles any framing required for the display session user to view all the frames in a multi-frame table report after the data reporter completes its processing.

The data reporter you code can generate either a row report or a table report. The maximum number of header lines is two.

A row report consists of one or two header lines and a single data line. For a row report, RMF invokes the data reporter twice: once to format the header line(s) and once to format the data line. When a row report is executed repetitively, RMF invokes the reporter to format the header line(s) for the first execution; for all subsequent executions, the reporter is invoked to format a data line.

A table report consists of one or two header lines and a variable number of data lines. For a table report, RMF invokes the data reporter once to format both the header line(s) and the data lines. The number of data lines must be less than or equal to the number of relocate blocks created in the SMF record by the data gatherer.

Upon entry to the data reporter, register 1 points to a contiguous list of eleven addresses that point to eleven input parameters. The first address points to the first parameter, the second address points to the second parameter, and so forth. The input parameters are:

First Parameter: A full word entry code that can be either X‘1’ or X‘2’. X‘1’ indicates that the reporter is to format the header line(s) for a row report. X‘2’ indicates, for a row report, that the reporter is to format the single data line. For a table report, the entry code should always be X‘2’, indicating that the reporter is to format both the header line(s) and the data lines.

Second Parameter: A full word report mode indicator that can have either of the following values:

X'1' Total mode; the values in the report are to reflect session totals.

X'2' Delta mode; the values in the report are to reflect changes since the last request for the report.

Third Parameter: The operands, if any, specified by the report user when he requested the report, in the form:

LL	text

LL A two byte length field indicating the length of the following text (does not include the two bytes of LL).

text A character string of up to 32 characters containing the report operands.

When the report has no operands or the report request did not include operands, LL is set to zeros.

Fourth Parameter: The default operands from ERBFMENU or ERBBMENU, in the form:

LL	text

LL A two byte length field indicating the length of the following text (does not include the two bytes of LL).

text A character string of up to 32 characters containing the default operands.

When the report has no operands or no default operands, LL is set to zeros.

Fifth Parameter: The address of the current SMF record buffer; that is, the buffer where the data gatherer has placed the data for the current execution of the reporter.

Sixth Parameter: The address of the previous SMF record buffer; that is, the buffer where the data gatherer placed the data for the previous execution of the report. When the report mode (the second parameter) indicates delta mode, the data fields in the previous SMF record enable your data reporter to calculate the changes that have occurred since the last request for the report.

Seventh Parameter: The first of the two words reserved for the use of your routines.

Eighth Parameter: The second of the two words reserved for the use of your routines.

Ninth Parameter: A byte containing the number of the subpool to use when you issue a GETMAIN to obtain the storage your routine requires.

Tenth Parameter: The address of the RMF putline routine. When the data reporter has formatted a report line, it calls the putline routine to perform the actual output operation.

Eleventh Parameter: The control block address that your data reporter must pass to the putline routine.

The processing your data reporting routine performs is determined largely by the nature of the report for which you are formatting report lines. This processing should include a validation of the entry code. If it is not a valid code, set a return code of 8 in register 15 and return control. If your report is a row report, examining the entry code determines whether your routine has been invoked to format the header line(s) or the data line for the report.

If the report has operands that can be specified when the report is requested, check the third input parameter to determine if the request specified operands. If it did, validate the syntax of the operands; if the syntax is invalid, set a return code of 4 in register 15 and return control. If the request did not specify operands, verify the syntax of the menu default operands passed as the fourth input parameter; if the syntax is invalid, set a return code of 24 in register 15 and return control.

If your report contains fields that are affected by the session mode – either delta mode or total mode – check the second input parameter to determine which mode is in effect. When delta mode is in effect, use the data fields in the previous SMF record buffer (pointed to by the sixth parameter) and the data fields in the current SMF record buffer (pointed to by the fifth parameter) to calculate the changes that have occurred since the last report request.

When your routine has formatted a report line, it should invoke the RMF putline routine to perform the actual output operation. To use the putline routine, perform the following steps:

1. Set up the input parameters that the putline routine requires. To do this, set register 1 to point to a list of four addresses that point to the following four parameters:

   First Putline Parameter: The record you have formatted, preceded by a two-byte length field. The length specified **must not** include the two bytes of the length field. The maximum record length is 79 characters. Note that the 3270 field attribute bytes must **not** be included; RMF supplies these bytes.

   Second Putline Parameter: A two-byte field that tells the putline routine whether the record you have formatted is a header line or a data line. The field must contain one of the following:

   'HD' Indicates that the record is a header line

   'DT' Indicates that the record is a data line

   Header lines generally contain column headings. These lines are repeated when the terminal user frames forward through a multi-frame table report or when the hardcopy output crosses a page boundary.

   Third Putline Parameter: A one-byte field; its bits have the following meaning:

Bit	Meaning
0	Set to 1 if high intensity display is desired. Set to 0 if low intensity display is desired. (The bit is ignored during a background session.)
1-7	Reserved. These bits must be set to zeros.

   Fourth Putline Parameter: The control block address that RMF passed to your data reporter in the eleventh input parameter.
2. Invoke the putline routine using standard linkage conventions. Set register 13 to point to your register save area, set register 15 to the address of the putline routine (passed to your data reporter in the tenth parameter), and pass control to the putline routine by a BALR 14,15 instruction.
3. When the putline routine returns control to the data reporter, a return code is set in register 15. A return code of zero indicates successful completion. A return code of 4, indicates an uncorrectable I/O error; set a return code of 12 in register 15 and return control.

When your data reporter has finished processing, set a return code in register 15 and return control by branching on the contents of register 14. Table 14 shows the possible return codes, their meaning, and the action RMF takes in response to each code.

Installing a user report

Once your data gatherer and data reporter are coded, two steps are required to install the report:

1. Include an entry for the report in the option list for a background session (ERBBMENU) and the menu list for a display session (ERBFMENU), depending on the type of session during which your report can be run.

   If data collected during a Monitor II background session is to be reported during execution of the Postprocessor, a copy of the option list control section (ERBBMENU) that includes the entry for your report must be link edited with the Postprocessor.

   RMF supplies the PICTURE macro to simplify the process of adding or changing an entry in the option list or menu list. See "Using the PICTURE macro" on page 153. You can also superzap an entry to make changes when the length of the entry is not changed.
2. Link edit your data gatherer and data reporter and test your report.

The option list or menu list consists of a set of variable-length entries, each describing a valid report. The option list appears in the RMF control section ERBBMENU; the menu list appears in ERBFMENU. Two separate control sections are provided to allow for a report that will run only during a background session or only during a display session. Also, the two different control sections allow different sets of default operands to be established for display sessions and background sessions. For example, you might want the display defaults to specify a limited set of possible data, while the background defaults specify all possible data.

The steps required to add an entry to the list are:
1. Determine whether the USER entry supplied by RMF is appropriate for your report. The USER entry contains specifications for a table report (RPTTYP=T) with a single relocate block (MAXRBS=1) that is four bytes long (RBLEN=4). The report title is 'USER PICTURE'. If the entry is not appropriate for your report, replace the entry with a new entry for USER.
2. If you are changing the USER entry or adding a new entry, make a copy of ERBFMENU for a display report or ERBBMENU for a background session — or both — from the source code data set.
3. In the copy you have made, either replace the USER entry or insert a new PICTURE macro. For a new display report, insert the PICTURE macro where you want the new report to appear in the menu frame. For details, see "Using the PICTURE macro" on page 153.
4. Assemble ERBFMENU for a display report and ERBBMENU for a background report.
5. Link edit the menu list or option list CSECT(s) that you have assembled into the RMF load modules:
   `ERBMFMFC` - RMF control
   `RMFMON` - Monitor II RMFMON command
   `ERBRMFPP` - Postprocessor
   `ERB2RCTL` - Monitor II ISPF version
   `ERB2XDG0` - Monitor II Internal Data Gatherer

   A sample of the control statements required is:

```
//LINKEXIT JOB   MSGLEVEL=1
//LINK0001 EXEC PGM=IEWL,PARM='MAP,XREF,REUS,RENT,REFR,NCAL'
//SYSPRINT DD    SYSOUT=A
//SYSLMOD  DD    DSN=SYS1.SERBLINK,DISP=(OLD,KEEP)
//SYSUT1   DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(TRK,(20,5))
//SYSLIN   DD    *
  (ERBFMENU object deck)
  (ERBBMENU object deck)
  INCLUDE SYSLMOD(ERBMFMFC)
  ENTRY ERBMFMFC
  ALIAS ERBMFMPR
  ALIAS ERBMFCLS
  SETCODE AC(1)
  NAME ERBMFMFC(R)
  (ERBFMENU object deck)
  INCLUDE  SYSLMOD(ERBMFTSO)
  ENTRY    ERBMFTSO
  ALIAS    RMFMON
  NAME     ERBMFTSO(R)
  (ERBBMENU object deck)
  INCLUDE  SYSLMOD(ERBRMFPX)
  ENTRY    ERBRMFPP
  NAME     ERBRMFPX(R)
  (ERBFMENU object deck)
  INCLUDE SYSLMOD(ERB2RCTL)
  ENTRY ERB2RCTL
  NAME  ERB2RCTL(R)
  (ERBFMENU object deck)
  INCLUDE SYSLMOD(ERB2XDG0)
  ENTRY ERB2XDG0
  NAME  ERB2XDG0(R)
/*
```

Figure 26. Install User Report

To install your report, you must link edit your data gatherer and data reporter.

If you are using the USER entry, name your gatherer routine ERBGUS99; name your reporter routine ERBRUS99. Replace the dummy RMF modules that have these names with your own routines. The link edit control statements required are:

```
(ERBGUS99 object deck)
ENTRY ERBGUS99
NAME ERBGUS99(R)
(ERBRUS99 object deck)
ENTRY ERBRUS99
NAME ERBRUS99(R)
```

If you are not using the USER entry, give your data gatherer and data reporter modules names that match the names you are specifying in the PICTURE macro for the report that you are adding. Link edit the modules as shown in the above control statements, replacing ERBGUS99 with the name of your data gatherer and ERBRUS99 with the name of your data reporter.

Once your modules have been link edited, you are ready to test your report. You might find it simpler to test your new report on TSO before making it available to other RMF users at your installation. Perform the following steps:

1. Use a testing tasklib, a special partitioned data set (for example, TESTLIB.LOAD). Place your data gatherer, data reporter, and the RMFMON load module that includes the new menu list in the testing tasklib.
2. You can then test the new report by entering:
   ```
   CALL TESTLIB(RMFMON)
   ```

The new menu should appear on the screen in response to this command. You can then invoke your report by specifying its menu item name.

If your report routine terminates abnormally, you can obtain a dump by replying 'STOP' to the messages describing the abnormal termination.

Using the PICTURE macro

The PICTURE macro describes a Monitor II session report to RMF. Use the PICTURE macro to replace the USER description or add or replace any entry in either ERBBMENU or ERBFMENU. The PICTURE macro is located in SYS1.MACLIB.

The syntax of the macro and the meaning of each operand are as follows:

```
[label]  PICTURE
         ID=name,
         GATHER=gathername,
         REPORT=reportname,
         RBLEN=length,
         RPTTYP={R|T}
         [,PFK=n]
         [,TITLE='title']
         [,DGTEXT='dgdefaults']
         [,DRTEXT='drdefaults']
         [,MAXRBS=nn]
         [,FBLEN=len]
         [,HELP={'*'|'panelname'}]
```

Figure 27. Syntax of the PICTURE Macro

ID=name
: The option or menu item that will identify the report.

  The name must consist of one to eight alphameric characters. The first character must not be 'R'; RMF takes 'R' to be a request to recall a report. For a display report, this name will appear on the menu frame.

GATHER=gathername
: The name of the module RMF is to invoke to gather data for the report.

PFK=n
: The PF key number associated with the report, where n is a one-digit or two-digit decimal identifier in the range of 1 to 24. For a display report, this number appears in the menu frame. If a PF key is not specified, the report is not associated with a PF key.

REPORT=reportname
: The name of the module RMF is to invoke to format the header lines and data line(s) for the report.

RBLEN=length
: The length of the relocate block generated by the data gatherer for each line in the report.

RPTTYP={R|T}
: The type of report. T indicates a table report; R indicates a row report.

TITLE='title'
: An optional report title. The title specified appears in the menu frame for a display session. The title must be enclosed in single quotation marks. Use a

double quotation mark to represent any quotation mark used in the title. The title can contain up to 50 printable characters. However, a maximum of 35 characters can be printed or displayed; therefore, a title longer than 35 characters will be truncated to fit into the menu frame.

DGTEXT='dgdefaults'
The default operands that are passed to the data-gathering routine for the report. This field is optional; it is used when the report requires operands. The text must be enclosed in single quotation marks, and the maximum length of the text is 32 characters. Any characters are valid between the quotation marks. Use two quotation marks to represent any quotation mark used in the text. When more than 32 characters are specified, the text is truncated.

DRTEXT='drdefaults'
The default operands that are passed to the data-reporting routine for the report. This field is optional; it is used when the report requires operands. The text must be enclosed in single quotation marks. Use two single quotation marks to represent any quotation marks used in the text. Any characters are valid between the quotation marks. When more than 32 characters are specified, the text is truncated.

MAXRBS=nnn
The initial number of relocate blocks. This number is equivalent to the maximum number of data lines in the report. The field is optional; when it is omitted, the default is 1 when RPTTYP=R is specified, indicating a row report. When RPTTYP=T is specified, indicating a table report, the field defaults to zero; however, enough storage is provided to allow a relocate block for each address space possible in the system. The maximum value possible for MAXRBS is 32,767.

FBLEN=len
The total length of all data control sections of the SMF record. The default value is 0.

HELP={'*'|*'panelname'*}]
Name of ISPF panel (maximal 8 characters) that contains help for this report. If HELP is requested on this report during a Monitor II ISPF display session, the panel *'panelname'* will be shown, if there is no message pending. If this option is omitted, '*' is generated by default which causes the tutorial displayed in such a case. The option has no effect for the TSO RMFMON session and for background sessions.

Except of **GATHER, REPORT, TITLE, DGTEXT, DRTEXT,** and **HELP**, all options are ignored, if the current picture is the second definition for a report with the same ID.

Example

The following example shows how to use the PICTURE macro to add a menu item to ERBFMENU. The menu item for the report is ANL, the data gatherer is ANLDG, the PF key is 23, the data reporter is ANLRP, the length of the relocate block is 32, the length of all data control sections is 0, and the report is a table report. The title of the report is USER ANALYSIS, the default operands for the gatherer and the reporter are 1,1,1. The maximum number of relocate blocks is 128.

```
ANLPIC PICTURE ID=ANL,GATHER=ANLDG,PFK=23,REPORT=ANLRP,RBLEN=32,FBLEN=0,
               RPTTYP=T,TITLE='USER ANALYSIS',DGTEXT='1,1,1',
               DRTEXT='1,1,1'MAXRBS=128
```

TSO terminal user authorization

All the data collected and reported by RMF during a Monitor II TSO display session is obtained from commonly addressable storage that is not fetch protected. However, if your installation wants to limit the use of the command that starts an RMF Monitor II (RMFMON) session under TSO, one method available is to replace the RMF control section with your own module. For Monitor II you replace the control section ERBTSOCK. Your routine will then be invoked as part of the RMF response to the RMFMON command.

Note: You cannot protect the ISPF session by ERBTSOCK. Instead, RACF® services should be used in order to prevent from unauthorized calling of RMF Monitor II.

ERBTSOCK (Monitor II) runs in problem state with a key of 8. When this control section gets control, register 1 points to a two-word address list. The first address points to the seven-byte userid of the user who has issued the RMFMON command. The second word points to the PSCB. Figure 28 illustrates the input parameter structure.

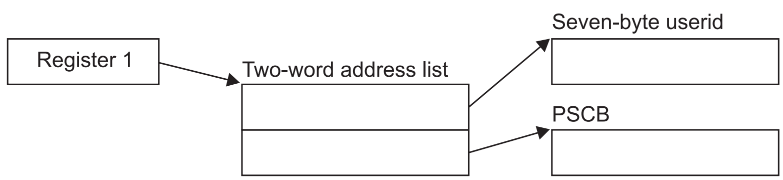


Figure 28. ERBTSOCK Input Parameter Structure

The module that you code to replace ERBTSOCK must be reenterable. It receives control by a BALR instruction and must save the registers when it receives control and restore the registers when it returns control. Register 13 contains the address of the register save area; register 14 contains the return address; and register 15 contains the entry address.

The processing your module performs depends on the method you choose to validate the user. Possible methods include issuing a RACHECK, prompting the user for a password, or checking the userid against a list of valid userids. Information on the TSO services available to perform these functions, such as TGET or TPUT, can be found in *z/OS TSO/E Programming Services*.

You can also use the PSCB bits defined for user use. This field (PSCBATR2 in the PSCB) comes from the UADS and can be updated by the USERDATA keyword of the ADD and CHANGE subcommands of the ACCOUNT command. See *z/OS TSO/E System Programming Command Reference* for more information on these commands.

TSO/E must be installed on your system to use the ACCOUNT, TGET and TPUT commands.

When your routine has completed its processing, set a return code of 0 in register 15 to indicate to RMF that the user is authorized to issue RMFMON. Set a return code of 4 in register 15 to indicate to RMF that the user is not authorized to issue RMFMON. In response to this return code, RMF displays a message to the terminal, and does not start the session. After setting the appropriate return code, RMF returns control by branching on the contents of register 14.

For the Monitor II TSO/E display session the user authorization exit routine (ERBTSOCK) is part of the RMF load module that contains the RMFMON command. This module resides in SYS1.SERBLINK as load module RMFMON; its entry point is ERBMFTSO. Before your authorization routine can execute, you must link edit it with RMFMON; the control statements required are:

```
(ERBTSOCK object deck)
INCLUDE ddname(RMFMON)
ENTRY ERBMFTSO
NAME RMFMON(R)
```

End of Programming Interface information

Chapter 6. Adding Monitor III user exits

RMF provides user exits to allow you to tailor data collection and reporting to the needs of your installation. There are three main advantages to this. You can:

- Add information to a standard Monitor III report
- Sort the information in a standard report in a different order
- Create new reports combining the data that Monitor III gathers in the way you need them

In principle, you can modify any Monitor III report, with the exception of the Group Response Time report.

Overview

Programming Interface information

The **RMF Monitor III Utility** (see "The Monitor III utility" on page 164) is the most important tool at your disposal for writing user exits. It is dialog-driven, and helps you use the necessary ISPF table services and RMF data-retrieval interface. However, you should be familiar with ISPF, Dialog Management Services, and RMF if you want to create and implement your own exit routines.

Data gathering

RMF generates Monitor III data by invoking a data gatherer module at each CYCLE. Replace the RMF dummy module ERB3GUSR with your own data gatherer routine, to have RMF invoke it, too, at each CYCLE.

Reporting

RMF takes several different actions in the course of producing a report, and the user exits allow you to modify each of these actions in order to change a report or produce a new one.

In the four separate processing **phases** of the reporter session, RMF:

1. Generates
2. Modifies
3. Formats and displays
4. Cleans up

the ISPF tables with the report data. The Monitor III Utility helps you to modify phases 1 and 3. Phases 2 and 4 are provided specially for user reports. See "Data reporter phases" on page 163 for more details.

Invoking user reports

The Monitor III Utility allows you to tailor RMF reports and to define the layout of new, user reports. RMF selects existing reports using ISPF SELECT, and uses the same method to select user-defined reports. To take advantage of this handling for your user reports:

- Use the Monitor III Utility to update the user-report selection panel
- Update the RMF command table, using the standard ISPF function

You can choose the time range to invoke the data reporter either:

- Before entering your user exit, by using the BREF/FREF commands or the RANGE/REFRESH session options
- Or from within the first phase of your reporter, by invoking the Data Retrieval Service module, ERB3RDRS, either by calling it or using the ISPF SELECT service.

"Data retrieval service (ERB3RDRS)" on page 192 describes this process.

Measurement data

The data gatherer collects data, and the data reporter uses this data to generate a formatted report for printing or display. The data gatherer module and the data reporter module communicate through control blocks that contain data from a set-of-samples.

Your user exits can use this means of communication, too. The format of the sample data is described in "Data gatherer sample structure."

End of Programming Interface information

Data gatherer sample structure

Programming Interface information

RMF writes *resource data records* with the data that the gatherer routine collects at each CYCLE, and combines them into a *sample*. At the end of each MINTIME period, RMF combines these samples into a *set-of-samples* in the data gatherer's address space, and moves the sets-of-samples into an in-storage buffer. The data reporter retrieves the data from this storage area, reduces it, and formats it for output.

Figure 29 shows the layout of three data areas that are common to all Monitor III data gatherers, whether coded by a user or provided by RMF. These areas are:

- The set-of-samples header
- The sample header
- The resource data record (RED)

Field offsets in the sample header and resource data record refer to offsets from the start of the control block containing the field. For example, the address of the first user record is the address of the REDG3 plus the offset to the first user record. All of these areas are maintained by RMF, specifically by the mainline data gathering module (ERB3GMFC). Figure 29 also shows the relationship between the data collected by the data gatherer user exit routine and the sample structure maintained by RMF.

Note: For a description of how Monitor III maintains a set of samples when VSAM data sets are used with data set support, see Chapter 7, "Using Monitor III

VSAM data set support," on page 197.

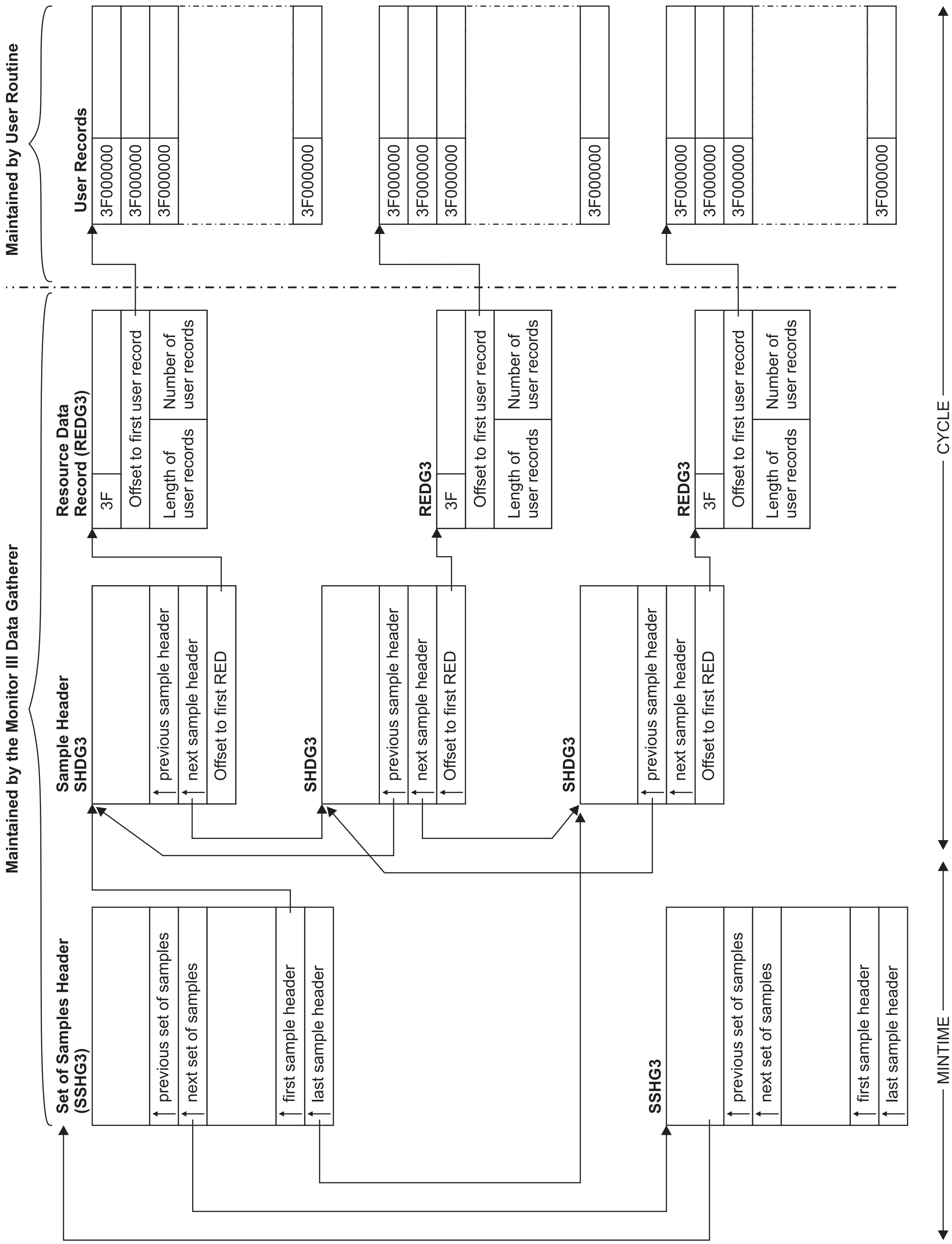


Figure 29. Data Gatherer Sample Structure

Data gatherer control blocks

Figure 30 describes the fields in the set-of-samples header control block, the sample header, and the resource data record. These data areas are mapped by the RMF macros ERBSSHG3, ERBSHDG3, and ERBREDG3.

```
                         SET OF SAMPLES HEADER
**********************  (ERBSSHG3 MAPPING MACRO)  ********************

SSHG3      DSECT            SAMPLE HEADER
           DS      0D       ALIGN ON DWORD BOUNDARY
SSHSSHG3   DS      XL5      ACRONYM SSHG3
SSHRMFV    DS      XL1      SSHG3 CONTROL BLOCK VERSION '05'X
SSHLEN     DS      H        LENGTH OF SSHG3
SSHRMFVN   DS      XL3      RMF VERSION NUMBER
SSHFLAG1   DS      XL1      FLAG BYTE
SSHGCOMP   EQU     X'80'    ON = DATA ARE COMPRESSED
SSHPREVP   DS      A        POINTER TO PREVIOUS SSH
SSHNEXTP   DS      A        POINTER TO NEXT SSH
           DS      4F       RESERVED
SSHSHDFP   DS      A        POINTER FIRST SAMPLE HEADER
SSHSHDLP   DS      A        POINTER TO LAST SAMPLE HEADER

                             SAMPLE HEADER
**********************  (ERBSHDG3 MAPPING MACRO)  ********************

SHDG3      DSECT            SAMPLE HEADER
           DS      0F       ALIGN ON WORD BOUNDARY
SHDSHDG3   DS      XL5      ACRONYM 'SHDG3'
SHDRMFV    DS      XL1      SHDG3 CONTROL BLOCK VERSION NUMBER X'02'
SHDLEN     DS      XL1      LENGTH OF SHDG3
SHDFLAG1   DS      XL1      SAMPLE FLAG 1
SHDINVAL   EQU     X'80'    SAMPLE IS INVALID
SHDPREVP   DS      A        POINTER TO PREVIOUS SAMPLE
SHDNEXTP   DS      A        POINTER TO NEXT SAMPLE
SHDREDOF   DS      A        OFFSET TO FIRST RED RECORD

                         RESOURCE DATA RECORD
**********************  (ERBREDG3 MAPPING MACRO)  ********************

REDG3      DSECT            RESOURCE RECORD
           DS      0F       ALIGN ON WORD BOUNDARY
REDREDID   DS      XL1      RED ID
REDUSRCB   EQU     X'3F'    RED ID FOR USER EXIT
REDFLAG1   DS      XL1      RED FLAG1
REDINVAL   EQU     X'80'    USER EXIT DATA ARE INVALID FOR THIS SAMPLE
REDRETRY   DS      H        NR OF RETRIES OF THE USER EXIT ROUTINE
REDFUWDO   DS      F        OFFSET TO FIRST USER EXIT RECORD
REDUSERL   DS      H        LENGTH OF USER EXIT RECORD
REDUSERN   DS      H        NUMBER OF USER EXIT RECORDS
```

Figure 30. Mapping Macros of ERBSSHG3, ERBSHDG3 and ERBREDG3

Set of samples header control block (SSHG3)

The set-of-samples header control block represents all samples collected during a MINTIME interval. This control block contains pointers to the previous and next set-of-samples header control block, as well as pointers to the first and last sample header control blocks. A set-of-samples is the smallest amount of data that the data reporter can retrieve. RMF maintains and updates all fields in this control block as needed.

Sample header control block (SHDG3)

This control block identifies a single sample taken at the end of a CYCLE. RMF identifies each sample with a sequence number and increments the sequence number at every CYCLE. This sample header contains forward and backward pointers to other sample header control blocks in the chain, as well as a pointer to the resource data record. RMF maintains and updates all fields in this control block as needed.

Resource data record (REDG3)

There is one resource data (RED) record for each defined resource in the system. RMF maintains and updates all fields in this record as needed. RMF uses RED records to access USE/WAIT records (in the case of the Monitor III data gatherer) or user records (in the case of a data gathering user exit routine). RED records are fixed in length, and contain X'3F' in the resource identifier (REDREDID) field when RMF invokes your data gatherer user exit routine. RMF uses this identifier to locate your user records, which also must have the same hexadecimal identifier. The RED record also contains the offset to the first user record (REDFUWDO), the length of your user exit records (REDUSERL), and the number of user exit records (REDUSERN) created during a CYCLE. While RMF maintains all the fields in the RED record, it obtains the length and number of user records from values you provide in the interface area used by the Monitor III data gatherer and your user routine. When RMF invokes your user exit, the second input parameter points to this interface area (see "Programming a data gatherer").

User record

A user record contains the information your data gathering routine collects at each CYCLE. The user record must be fixed in length and the first four bytes must contain the identifier X'3F000000'. You define the remaining fields in the user record and fill them in with the data you collect. The format of the data in the user record depends on the report you are generating. You set the format that best meets your needs.

End of Programming Interface information

Programming a data gatherer

Programming Interface information

The data gatherer runs in the Monitor III data gatherer address space in problem state, with a key of 8, and in 31-bit addressing mode. The data gatherer must be coded as reentrant. It receives control by a BALR instruction and must save the registers when it receives control and restore the registers when it returns control. The register contents are:

Register 13 Address of the register save area
Register 14 Return address
Register 15 Entry address

Upon entry to the data gatherer, register 1 points to a contiguous list of three addresses that point to three input parameters. The first address points to the first parameter, the second address points to the second parameter, and the third address points to the third parameter. The input parameters are:

First Parameter

An area containing the management fields for the Monitor III data gatherer and the user data gatherer exit routine. The GGDMODAR DSECT (global data gatherer control block) is mapped by the ERBGGDG3 macro and describes the dynamic storage obtained when your data gatherer routine issues the GETMAIN macro. When RMF invokes your routine for the first time, it provides information in the following fields:

GGDMODNA
The module name, which is ERB3GUSR.

GGDAUSBP The subpool number from which your routine must obtain storage via the GETMAIN macro.

GGDREDID The resource identifier, which is X'3F'.

You must fill in the address and the length of the storage area (within the user subpool) that you obtain with the GETMAIN macro. The Monitor III data gatherer can then free this area at the end of the gatherer session. The fields in the global data gatherer control block that you must fill in are:

GGDAULEN The length of the storage area.
GGDAUPTR The address of the storage area.

All other fields in the GGDMODAR control block are set to zeroes. The contents of the fields in GGDMODAR are not changed by RMF between calls to your user exit routine.

Second Parameter
The interface area between the Monitor III data gatherer and the user exit routine. The interface area is reinitialized by RMF before each call to the exit. The interface area is four fullwords in length and contains the following:

- First fullword – The user subpool number from which the user exit routine must obtain storage via the GETMAIN macro if additional storage is required.
- Second fullword – The address of the retry work area (RETSTACK DSECT) used in error recovery. The ERBGGDG3 macro maps this retry work area. RMF provides this address, and your routine must not destroy it. The RETSTACK DSECT contains information that the Monitor III data gatherer error recovery module (ERB3GESA) uses if an error occurs in your data gatherer exit routine. Because RMF provides a recovery environment, it is not necessary to provide an ESTAE exit for your routine. If you choose to use the ESTAE or SPIE macro, you must not alter the Monitor III error recovery environment. You might choose to have your exit routine get control as a retry routine in the event of an abend. For example, if a control block chain changes while your data gatherer routine is scanning it, then your exit routine might abend. In this case, you must set up several fields in the retry work area at each invocation of your user exit routine, so that the Monitor III data gatherer can return control to your routine. These fields are:

  RETADDR Contains the retry entry point address in your routine. The data gatherer returns control to the user exit routine at this address when attempting to retry after an error. In cases where the number of retries is exhausted, the error recovery module (ERB3GESA) returns control to the main data gatherer module (ERB3GMFC) and not the data gatherer exit routine.

  RETCOUNT Contains the number of times the user exit routine can be retried during one invocation. The RMF error recovery routine decrements the number in this field each time it gets control.

  RETRUBFL Specifies registers that must be restored by the recovery termination manager (RTM) before returning control to the address in your routine specified in the RETADDR field.

This field should contain X'FFFF', indicating that all registers must be restored after must be restored after error-recovery processing completes.

RETREGSA A 16-word storage area used to store the contents of the registers specified in the RETRUBFL field.

- Third fullword – The address of an area containing the data the user exit routine collects. Your routine must supply this address each time it is invoked. RMF uses this address to move the collected data from the exit routine's storage area into the data gatherer's in-storage buffer.
- Fourth fullword – Two halfwords that the user exit routine must provide at each invocation. The first halfword must contain the length of the user record, and the second halfword must contain the number of user records collected during the current cycle. RMF places the length and number of user records in the resource data (RED) record. All user records must be fixed in length and must start with a fullword hexadecimal identifier of X'3F000000'. RMF uses this information to move your collected data into the in-storage buffer.

Third Parameter

The address of the return code of the user exit routine.

If your user exit routine successfully gathers all the data needed for your report, set a return code of X'00' in the area pointed to by this parameter in the parameter list. RMF will invoke your user exit routine at the next CYCLE. If you do not want RMF to invoke your routine again, set a return code of X'10'. Return to the caller by branching on the contents of register 14.

The processing your data gathering routine performs depends largely on the nature of the report for which you are gathering data. The first time RMF invokes your data gatherer routine, it provides a subpool number (in the GGDAUSBP field) that you must use when issuing the GETMAIN macro. After issuing a GETMAIN for the dynamic storage it needs to execute in, your routine must place the address and length of the storage obtained in the GGDAUPTR and GGDAULEN fields, respectively. (The GGDAUPTR and GGDAULEN fields contain zeroes when RMF invokes your routine for the first time.) When RMF makes subsequent calls to your routine, these two fields still contain the address and length of your dynamic storage. You do not have to issue another GETMAIN and you can reuse the storage obtained on the first call. This function eliminates the overhead of issuing a GETMAIN for dynamic storage each time RMF invokes your routine. Depending on the amount of data you collect, you may need to obtain additional storage to hold your user records.

End of Programming Interface information

Data reporter phases

Programming Interface information

To display a user-modified or user-created report, RMF makes use of ISPF tables that contain information about the report. You can control four phases to modify or create these tables and to generate and display your own reports for an RMF session.

Note: RMF uses two of these phases to generate and display standard RMF reports. Most of the unmodified standard reports, however, are not kept in ISPF tables. These tables are used primarily for user-modified and user-created reports.

The four phases and the activities performed in each are as follows:

- **Phase 1:** RMF generates an ISPF table that contains display data for every modifiable RMF report. Chapter 8, "Monitor III data reporter tables," on page 273 describes these tables. The time range for the display data for your routine can be changed during this phase by calling the Data Retrieval Service (ERB3RDRS) module. See "Data retrieval service (ERB3RDRS)" on page 192 for information about how to invoke the Data Retrieval Service.

  RMF does not use the Data Retrieval Service.
- **Phase 2:** RMF invokes your routine to allow you to modify the ISPF table generated in phase 1 in order to change an existing report or create a new report. RMF does not use this phase; you supply your own routine.
- **Phase 3:** RMF formats the ISPF table created in phase 1 or modified in phase 2 and displays the tabular or graphic version of the report through the ISPF service TBDISPL.
- **Phase 4:** RMF invokes your routine to allow you to perform various clean-up operations (for example, to free resources allocated for use in previous phases). RMF does not use this phase; you supply your own routine.

Note: If you decide to replace any of these phases, you must conform to the standards and externals described in this manual. If you do not, the results are unpredictable. See "Installing your own phases" on page 188.

End of Programming Interface information

The Monitor III utility

Programming Interface information

To help you with the steps outlined above, use the Monitor III report format definition utility. This utility consists of a series of ISPF panels that allow you to modify the ISPF tables that RMF uses during the four phases.

The three ISPF tables used to control RMF report formatting and display are:

- The phase driver table ERBPHDS3, which contains all RMF-supplied report definitions to generate reports during phase 1.
- The tabular report format table ERBFMTS3, which contains the information used to format each RMF tabular report during phase 3.
- The graphic parameter report table ERBPTGS3, which contains entries for the graphic version of each RMF report during phase 3.

Chapter 8, "Monitor III data reporter tables," on page 273 contains samples of each table and its entries.

You should be familiar with ISPF and TSO to use the report panel definition utility.

Report utility panel flow

Figure 31 shows the panel sequence for the report format definition utility.

To exit any panel, you can enter CANCEL on the command line or press END (PF3). If you enter CANCEL, the report format definition utility displays the report definition initialization panel (ERB3RD1) but saves none of your changes. If you press END on any panel, RMF displays the previous panel but does not save changes you have made. To continue viewing panels in sequence, press ENTER.

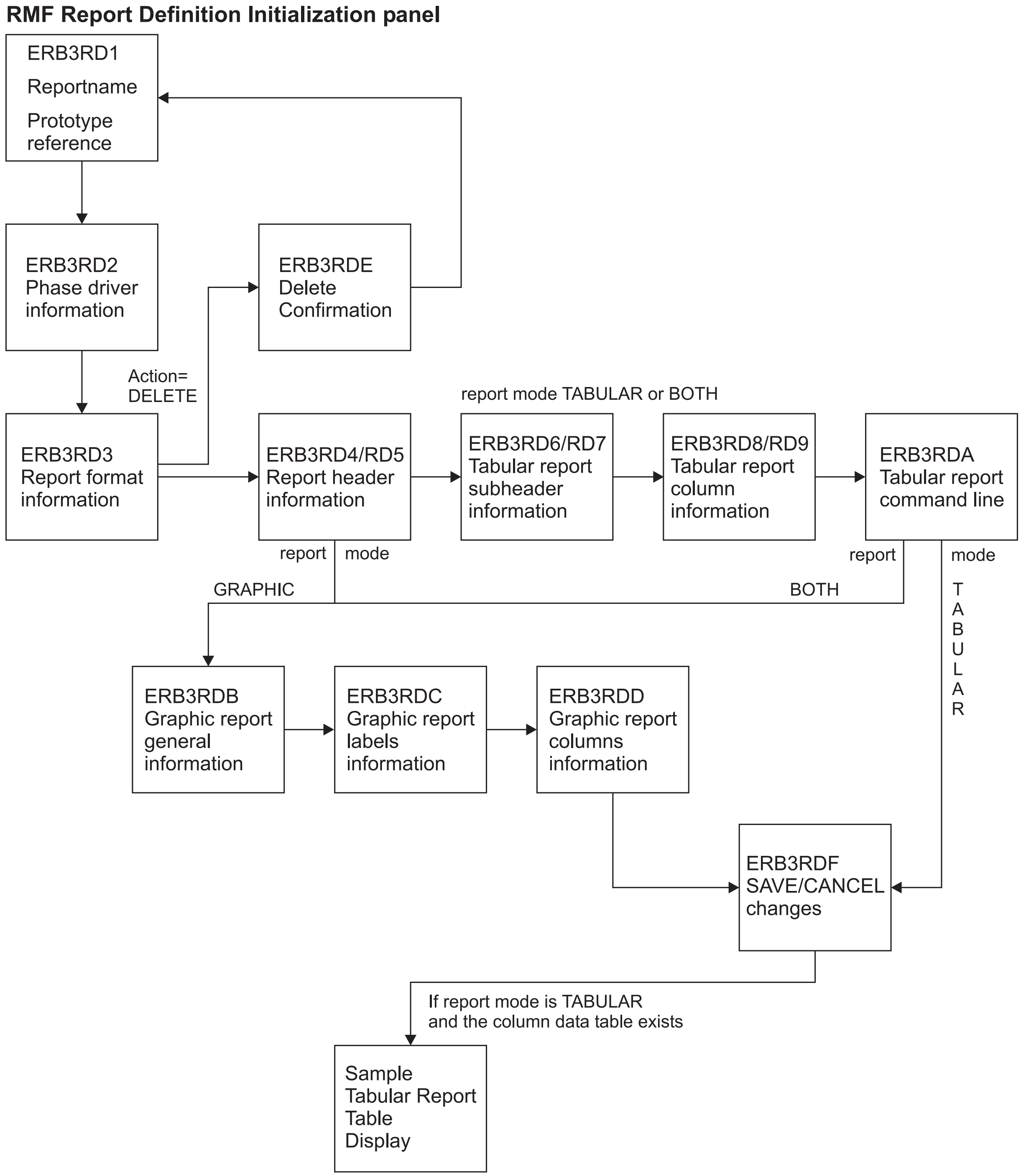


Figure 31. Panel Sequence for the Report Definition Utility

Before you start the utility

> **Note:**
> The actions described in this section are only required if you do not want to use the standard concatenation of the RMF libraries.

Do not use the RMF distribution table library as your ISPF output library (ERBTAB); you could destroy standard RMF report formats as a result. Allocate ERBTAB as part of a private user table library. You can concatenate this private library to the beginning of the RMF input table library (ERBTLIB) and can safely delete the ISPF tables you have modified or created (ERBPHDS3, ERBFMTS3, and ERBPTGS3) for your own reports.

You can merge your own libraries with RMF libraries. If you want to change the data set names and the allocations, modify CLIST ERBRMF3X. CLIST ERBRMF3X allocates the RMF ISPF libraries from the following distribution libraries:
- Panels from SYS1.SERBPENU
- Tables from SYS1.SERBTENU
- Messages from SYS1.SERBMENU

This CLIST is available in SYS1.SERBCLS, which must be concatenated to your SYSPROC library.

Starting the report utility

As a prerequisite for the invocation of the Monitor III report format definition utility, SYS1.SERBCLS must be concatenated to your SYSPROC library. For more information, refer to chapter "Setting up RMF" in the *z/OS RMF User's Guide*.

To start the utility, use one of the following commands:
- from TSO/E ready mode: `RMF UTIL`
- from within ISPF: `TSO RMF UTIL`

If you have the Kanji version of RMF, you start the Monitor III utility by entering:
`RMFJPN UTIL`

Notes:
1. Do not use a 3270 session with a screen size lower than 32x80.
2. Do not try to access the report format definition utility in split screen mode when you are in an active RMF Monitor III reporter session.

For more information about a specific panel, use the HELP keys.

Example - Modified SYSINFO report

The task how to create a new Monitor III report will be shown based on the example of a modified SYSINFO report. The SYSINFO report has this format:

```
                        RMF V1R13  System Information           Line 1 of 28
Command ===>                                                    Scroll ===> HALF

Samples:     60  System: SYSE  Date: 06/08/2011  Time: 08.59.00  Range:     60Sec

Partition:    SYSE    2084  Model314           Appl%:     22   Policy:STANDARD
CPs Online:   4.0     Avg CPU Util%:   24     EAppl%:     22  Date:   05/12/05
AAPs Online:  1.0     Avg MVS Util%:   26     Appl% AAP:  19  Time:   12.20.27
IIPs Online:    -                             Appl% IIP:   -
Group     T WFL --Users--  RESP TRANS -AVG USG-   -Average Number Delayed For -
            %   TOT  ACT  Time  /SEC PROC  DEV   PROC  DEV STOR SUBS OPER  ENQ

*SYSTEM      98  167    2        0.12  1.6  0.1    0.0  0.0  0.0  0.0  0.0  0.0
*TSO               3    0        0.12  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*BATCH       99    3    2        0.00  1.5  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*STC         91  158    0        0.00  0.1  0.1    0.0  0.0  0.0  0.0  0.0  0.0
*ASCH              0    0        0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*OMVS              3    0        0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
*ENCLAVE           0  N/A         N/A  0.0  N/A    0.0  N/A  0.0  N/A  N/A  N/A
BATCH     W  99    4    2  .000  0.00  1.5  0.0    0.0  0.0  0.0  0.0  0.0  0.0
BATCHLOW  S  99    3    2  .000  0.00  1.5  0.0    0.0  0.0  0.0  0.0  0.0  0.0
OMVSKERN  S        1    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
OMVS      W        2    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
OE        S        2    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
STC       W 100   26    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
STCDEF    S 100   26    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
SYSTEM    W  90  132    0  .000  0.00  0.1  0.1    0.0  0.0  0.0  0.0  0.0  0.0
SYSSTC    S 100  111    0  .000  0.00  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
SYSTEM    S  88   21    0  .000  0.00  0.1  0.1    0.0  0.0  0.0  0.0  0.0  0.0
TSO       W        3    0  .010  0.17  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
TSODEF    S        3    0  .010  0.17  0.0  0.0    0.0  0.0  0.0  0.0  0.0  0.0
BCP       R  99    3    2  .000  0.00  1.5  0.0    0.0  0.0  0.0  0.0  0.0  0.0
```

Figure 32. SYSINFO Report

The target is to create a report called SYSCPU that provides some more CPU related information as TCB% and SRB% for each group. This data is available in the corresponding Monitor III table.

```
                        RMF V1R13   System Information            Line 1 of 26
Command ===>                                                 Scroll ===> PAGE
Press END to return.
Samples:     60  System: SYSE  Date: 06/08/2011  Time: 08.59.00  Range:     60Sec

Partition:    SYSE    2084  Model 314         Appl%:      22  Policy: STANDARD
CPs Online:   4.0     Avg CPU Util%:   24     EAppl%:     22  Date:   05/12/05
AAPs Online:  1.0     Avg MVS Util%:   26     Appl% AAP:  19  Time:   12.20.27
IIPs Online:    -                             Appl% IIP:   -

Group     T WFL --Users--  RESP TRANS  CPU   TCB   SRB  -AVG USG-   -Avg Del-
              %   TOT  ACT  Time  /SEC   %     %     %   PROC  DEV   PROC  DEV

*SYSTEM      98  167    2        0.12  21.6  21.5   0.2  1.6  0.1    0.0  0.0
*TSO               3    0        0.12   0.0   0.0   0.0  0.0  0.0    0.0  0.0
*BATCH       99    3    2        0.00  20.5  20.5   0.0  1.5  0.0    0.0  0.0
*STC         91  158    0        0.00   1.2   1.0   0.2  0.1  0.1    0.0  0.0
*ASCH              0    0        0.00   0.0   0.0   0.0  0.0  0.0    0.0  0.0
*OMVS              3    0        0.00   0.0   0.0   0.0  0.0  0.0    0.0  0.0
*ENCLAVE           0  N/A         N/A   N/A   N/A   N/A  0.0  N/A    0.0  N/A
BATCH     W  99    4    2  .000  0.00  20.5  20.5   0.0  1.5  0.0    0.0  0.0
BATCHLOW  S  99    3    2  .000  0.00  20.5  20.5   0.0  1.5  0.0    0.0  0.0
OMVSKERN  S        1    0  .000  0.00   0.0   0.0   0.0  0.0  0.0    0.0  0.0
OMVS      W        2    0  .000  0.00   0.0   0.0   0.0  0.0  0.0    0.0  0.0
OE        S        2    0  .000  0.00   0.0   0.0   0.0  0.0  0.0    0.0  0.0
STC       W 100   26    0  .000  0.00   0.1   0.1   0.0  0.0  0.0    0.0  0.0
STCDEF    S 100   26    0  .000  0.00   0.1   0.1   0.0  0.0  0.0    0.0  0.0
SYSTEM    W  90  132    0  .000  0.00   1.0   0.9   0.2  0.1  0.1    0.0  0.0
SYSSTC    S 100  111    0  .000  0.00   0.3   0.3   0.0  0.0  0.0    0.0  0.0
SYSTEM    S  88   21    0  .000  0.00   0.7   0.6   0.2  0.1  0.1    0.0  0.0
TSO       W        3    0  .010  0.17   0.0   0.0   0.0  0.0  0.0    0.0  0.0
TSODEF    S        3    0  .010  0.17   0.0   0.0   0.0  0.0  0.0    0.0  0.0
BCP       R  99    3    2  .000  0.00  20.5  20.5   0.0  1.5  0.0    0.0  0.0
```

Figure 33. SYSCPU Report as Modification of the SYSINFO Report

You find details about all values that can be displayed for all Monitor III reports in Chapter 8, “Monitor III data reporter tables,” on page 273.

Report format definition panel (ERB3RD1)

After you call the report format definition utility by RMF UTIL, you get the **Report Definition Initialization** panel (ERB3RD1). On this panel, you can specify whether you want to create a new report or modify or delete an existing one. You can also select the name of an existing RMF report to use as a prototype for the new report.

```
ERB3RD1               RMF Report Format Definition              Row 1 of 7
Command ===> _

Enter the following information.  To continue press ENTER.
To exit enter CANCEL or press the END key.

ACTION          ===> CREATE         MODIFY, CREATE or DELETE
REPORT NAME     ===> SYSCPU         Name of report
  WLM MODE      ===> GOAL           WLM Mode of report (GOAL or COMPAT)

Enter following information only, if you want to use an existing report
definition as a prototype for the new report you want to create.

PROTOTYPE NAME  ===> SYSINFO        Name of existing report to be used
  WLM MODE      ===> GOAL           WLM Mode of existing report to be used

 The following report names are available for MODIFY or as prototype

 CACHDET    CACHSUM    CFACT      CFOVER     CFSYS      CHANNEL
 CPC        DELAY      DEV        DEVN       DEVR       DEVT
 DSD        DSINDEX    DSND       DSNJ       DSNV       ENCLAVE
 ENQ        ENQR       HSM        IOQ        JES        JOB
 MSI        OPD        PROC       PROCU      RG         RLSDS
 RLSLRU     RLSSC      SPACED     SPACEG     STOR       STORC
 STORCR     STORF      STORR      STORS      SYSENQ     SYSINFO
 SYSRTD     SYSSUM     SYSTREND   SYSWKM     WFEX       XCF
 ZFSACT     ZFSSUM
******************************* Bottom of data ********************************
```

Figure 34. Report Definition Initialization Panel ERB3RD1

The panel fields and their meanings are:

ACTION

Specifies the action you want RMF to perform as follows:
- MODIFY - to change an existing RMF report
- CREATE - to create a new report
- DELETE - to delete an existing report

REPORT NAME

Specifies the name of the report that RMF is to modify, create, or delete. The report name must conform to ISPF naming conventions.

WLM MODE

Specifies the mode of the report, either compatibility or goal mode.

PROTOTYPE NAME

When you enter CREATE for ACTION, specify the name of an existing RMF report to use as a prototype or model for your report. RMF provides you those report values, which you can change when you modify or create your report.

When you enter MODIFY or DELETE for ACTION, you can ignore this field.

Phase driver information panel (ERB3RD2)

Press ENTER to display the next panel, the **Phase Driver Information** panel (ERB3RD2).

On this panel, you can specify the selection character(s) to use for the new or modified report on the Primary menu of a report session. You can also specify for each reporter phase the program or CLIST to modify, create, or print your report, or perform clean-up services and routines.

If you want to modify an existing RMF report without changing the layout or header information, you can provide your own program or CLIST for phase 2 on this panel. You can use ISPF services and commands like TBSORT, TBDELETE, or TBCREATE to perform these modifications during phase 2.

If you want to modify an existing RMF report format or layout without adding or deleting lines from a report, you can specify the name of the RMF report you want to modify for phase 1 (optionally for phase 2) and the name of the standard program that RMF uses to format RMF reports for phase 3. See PHASE 3 STRING in Figure 35. You can then use the remaining report format definition utility panels to make the header and layout changes for the modified report.

If you want to create a report, you should use a prototype (see Figure 34 for the Report Format Definition panel) and make sure to include the report selection on the Primary menu for the RMF report session.

Figure 35 is an example of a Phase Driver Information panel that contains information about the SYSINFO report. It assumes that the new SYSCPU report will become available as option 4 in the User Selection menu.

```
ERB3RD2                 RMF Report Format Definition
Command ===>

Report Name: SYSCPU                          Section 1: Phase Driver Information
WLM Mode:    GOAL
             Definitions on this panel are independent of WLM mode.

Enter the following information.  To continue press ENTER.
To quit enter CANCEL. To go backwards press END.

Select Strings format is: PGM(nnnnnnnn) PARM(mmm) or CMD(nnnnnnnn mmm)

SELECTION CHARACTERS  ===> U.4        Selection on Primary Option Panel

PHASE 1 SELECT STRING ===> PGM(ERB3RPH1) PARM(SYSINFO)
        TABLE  NAME   ===> ERBSYST3   Name of reporter phase 1 result table

PHASE 2 SELECT STRING ===>
        TABLE  NAME   ===> ERBSYST3   Optional name of phase 2 result table

PHASE 3 SELECT STRING ===> PGM(ERB3RDSP)

PHASE 4 SELECT STRING ===>
```

Figure 35. Phase Driver Information Panel (ERB3RD2)

The panel fields and their meanings are as follows:

SELECTION CHARACTERS
: Specifies a 1 to 8 character alphanumeric value that RMF uses as a selection value on the Primary menu of a report session. You must have defined these selection characters in the menu panel.

  If you enter a selection that is currently used on the Primary menu of a report session, RMF displays the report that you modify or create on this panel when you make the selection.

PHASE 1 SELECT STRING
: Specifies the name of the program or CLIST that the reporter control module (ERB3RDPC) uses to generate the ISPF report table during phase 1. You must specify a CLIST for CMD or program for PGM. (Follow the rules for ISPF

SELECT services.) If you are modifying an existing RMF report or creating a new report using a prototype, you must specify for PGM the program name ERBRPH1, and for PARM the command name of the RMF report that you are modifying or using as a prototype. If you are creating a new report, be sure to include the report as a selection on the Primary menu or on the User Selection menu.

See the RMF supplied phase driver table (ERBPHDS3) in Chapter 8, "Monitor III data reporter tables," on page 273 for a list of the RMF program and PARM names.

PHASE 1 TABLE NAME
Specifies the name of the ISPF table that results when your program or CLIST is invoked during phase 1. You must specify this parameter if you have specified PHASE 1 SELECT STRING.

For a list of the RMF report data tables (PHDRTAB1) in the RMF supplied phase driver table (ERBPHDS3), see Chapter 8, "Monitor III data reporter tables," on page 273.

PHASE 2 SELECT STRING
Specifies the name of the program or CLIST used to modify the ISPF report data table created in phase 1. If you are creating a new report without having specified a prototype, you must enter the name of your CLIST to create the new report. (Follow the rules for ISPF SELECT services.) If you are modifying only the report header or layout of an existing RMF report, you do not need to enter a PHASE 2 SELECT STRING.

PHASE 2 TABLE NAME
Specifies the name of the ISPF table that results after phase 2. If you have entered a value for PHASE 2 SELECT STRING, you must specify a valid phase 2 table name.

If you are modifying the report header or layout of an existing RMF report, you can enter the same name you entered for PHASE 1 TABLE NAME.

PHASE 3 SELECT STRING
Specifies the program or CLIST that RMF uses to initiate phase 3 to format your report.

If you do not provide a program or CLIST for this field, RMF skips the remaining report format definition utility panels and displays the report definition initialization panel ERB3RD1. When you invoke your report during an RMF session, RMF does not display the report.

If you are creating a report and you want RMF to display it, specify PGM(ERB3RDSP), the standard RMF display module.

PHASE 4 SELECT STRING
Specifies the program or CLIST that ERB3RDPC uses to initiate phase 4. This field is optional.

Report format information panel (ERB3RD3)

If you have entered a name for PHASE 3 SELECT STRING on ERB3RD2, RMF next displays the **Report Format Information** panel (ERB3RD3). This panel is the first in a series of panels that allows you to change the header and subheader layout of an RMF report.

On this panel (ERB3RD3), you can specify tabular or graphic, or both the tabular and graphic displays for the report, the panel name of the tabular version of the report, or specify the name of a report help panel.

Figure 36 is an example of a Report Format Information panel for the SYSINFO report:

```
ERB3RD3                 RMF Report Format Definition
Command ===>

Report Name: SYSCPU                          Section 2: Report Format Information
WLM Mode:    GOAL

Enter the following information.  To continue press ENTER.
To quit enter CANCEL. To go backwards press END.


REPORT MODE          ===> BOTH         TABULAR, GRAPHIC or BOTH


PANEL NAME           ===> ERB4CPU      Name of tabular report panel


HELP PANEL NAME      ===> ERB4CPU0     Name of HELP panel


LOGICAL LINE NUMBER  ===> SYSDTLLN     Name of table variable
SEQUENCE NUMBER      ===> SYSDTPSN     Name of table variable
```

Figure 36. Report Format Information Panel (ERB3RD3)

The panel fields and their meanings are as follows:

REPORT MODE
Specifies the display mode for the report. Valid values are as follows:
TABULAR
GRAPHIC
BOTH

PANEL NAME
Specifies the name of the ISPF display panel for the tabular version of the report when you enter TABULAR or BOTH for REPORT MODE.

For a tabular report, you must specify the name of the display panel that is to contain the report information. RMF-supplied panel names that you can use are ERB3DSI (if you are modifying or using the DI screen as a prototype), ERB3SRR (if you are modifying or using the STORR delay report as a prototype), ERB3SYS (if you are modifying or using the SYSINFO report as a prototype), ERB3WFX (if you are modifying or using the WFEX report as a prototype), or ERB3CMN (if you are modifying or using any other report as a prototype).

If you specify the name of your own panel, make sure that the panel includes the following information:

- Output fields for 2 standard header lines (DSPHDR1 and DSPHDR2)
- Output fields for up to 5 subheader lines (DSPSUBH1 - DSPSUBH5) contained in the RMF report you want to modify

- Output fields for up to 3 column header lines (FMTCOLH1 -FMTCOLH3) contained in the RMF report you want to modify. For a description of the report format table ERBFMTS3, see Chapter 8, "Monitor III data reporter tables," on page 273.
- Up to 3 model line variables (FMTMODL1 - FMTMODL3) contained in the model section of the RMF report you want to modify. For a description of the entries in the report format table ERBFMTS3, see Chapter 8, "Monitor III data reporter tables," on page 273.
- The command line (defined by variable ZCMD) and scroll amount field (defined by variable AMT)

Also, ensure that the user-defined panel for your report includes an initialization (INIT), reinitialization (REINIT), and processing (PROC) section as in the RMF-supplied panels.

If you enter GRAPHIC for REPORT MODE, leave PANEL NAME blank.

HELP PANEL NAME
When you enter a value for PANEL NAME, specifies the name of the ISPF help panel that contains help information for your report. The field is optional.

LOGICAL LINE NUMBER/SEQUENCE NUMBER
Specifies the name of key variables in the data table of the RMF report you are modifying. A logical line number identifies a logical group of related data rows within a report; a line sequence number identifies each physical table row that belongs to the logical group.

The logical line number (that identifies the entire data group) is 1; the sequence number (the number of physical lines that belong to the logical group and include the volume serial/device type on one line and the space type on the second line of the graphic report) is 2 or more.

When you toggle between tabular and graphic reports, RMF uses these variables to synchronize the line or bar displayed on the screen (the beginning of a logical group of data table rows). For examples of RMF report data tables, see Chapter 8, "Monitor III data reporter tables," on page 273.

Report header layout panels (ERB3RD4 and ERB3RD5)

Press ENTER to display the next panel, ERB3RD4, the **Report Header Layout** panel.

Each RMF report contains report headings, subheadings, and columns that you can modify. The Report Header Layout panels (ERB3RD4 and ERB3RD5) allow you to change up to 2 header lines for the tabular and graphic versions of the report.

On the first of these panels (ERB3RD4), you can specify the header lines and header variables for your report. At the bottom of the panel, enter the header lines exactly as you want them to appear on your report. You can use the variables listed on the panel to appear in the headings of your report. (Panel ERB3RD4 lists variables from header data table ERBHDRS3. For the meaning of all variables in ERBHDRS3, see Chapter 8, "Monitor III data reporter tables," on page 273.)

If a variable name is too long to enter in the header line, you can use a placeholder (&Z). After you press ENTER, you define these placeholders with variable names on the next panel.

Figure 37 is an example of a report format definition panel ERB3RD4 that shows you the headings and variables for the SYSINFO report with the modified report title `CPU Information`:

```
ERB3RD4                 RMF Report Format Definition
Command ===>

Report Name: SYSCPU     WLM Mode: GOAL         Section 3: Report Header Layout

Enter or change the report header lines.  To continue press ENTER.
To quit enter CANCEL. To go backwards press END.

You may intermix: text, variables, and variable placeholders (&Z).
If you specify variable placeholders (&Z) the next panel will ask you
to specify the variable name that is to replace each &Z

The following variables are available for use in the header:
  &ERBSID     &ERBSAMPL   &ERBTIME    &ERBRMFVD   &ERBSNUM
  &ERBHCTXT   &ERBDATE    &ERBRANGE   &ERBSPXID   &ERBSAMWL

Variables ERBSID, ERBDATE, ERBTIME and ERBRANGE will be supported as input
fields only, if they are part of second header line.

Enter or change up to two report heading lines:

           &ERBHCTXT    &ERBRMFVD  CPU Information
Samples: &Z      System: &Z    Date: &ERBDATE  Time: &ERBTIME  Range: &Z    Sec
```

Figure 37. Report Header Layout Panel (ERB3RD4)

In Figure 37, two report header lines appear at the bottom of the panel and ten variable names are available for the header lines.

- Variables &ERBHCTXT and &ERBRMFVD are specified at the beginning of the first header line.
- Variables &ERBDATE and &ERBTIME are specified for `Date` and `Time`.
- Placeholders (&Z) for the other variables (&ERBSID for session id, &ERBSAMPL for samples, and &ERBRANGE for range) appear in the appropriate fields of the header lines and indicate that the variable names they represent might not fit in the space provided. These placeholders can be defined on the next panel.

Press ENTER to display the second **Report Header Layout** panel (ERB3RD5).

On ERB3RD5, you can specify variable names for any Z placeholders you have used. The headings, variables names, and placeholders as you entered them on ERB3RD4 appear at the top of the panel. The variable names appear under the headings in the order specified on ERB3RD4. You can specify your own variable names in the spaces provided; however, in order for RMF to display the user-specified variables during a report session, they must be in the function pool for phase 3 or in the shared ISPF variable pool. Otherwise, blanks appear in the report. See “Installing your own phases” on page 188.

You must specify a number for each Z placeholder and its corresponding variable. Numbers must start with 1 and continue in sequence. There must be a one-to-one correspondence between placeholders and variable names, each pair with a unique number assigned to indicate the order of placement of the variable.

Figure 38 is an example of Report Header Layout panel ERB3RD5 that defines the placeholders used on the previous panel. If you do not have placeholders to define, press ENTER to get the next panel.

```
ERB3RD5                RMF Report Format Definition
Command ===>

Report Name: SYSCPU                              Section 3: Report Header Layout
WLM Mode:    GOAL

The following report header lines have been specified:
           &ERBHCTXT    &ERBRMFVD  CPU Information
Samples: Z1      System: Z2    Date: &ERBDATE  Time: &ERBTIME  Range: Z3    Sec

Specify the placeholder (Z) number next to the variable name to replace each Z
above.To continue press ENTER.To go backwards press END.To quit enter CANCEL.


     &ERBSID   ===> 2           &ERBHCTXT ===> __          &ERBSAMPL ===> 1
     &ERBDATE  ===> __          &ERBTIME  ===> __          &ERBRANGE ===> 3
     &ERBRMFVD ===> __          &ERBSPXID ===> __          &ERBSNUM  ===> __
     &ERBSAMWL ===> __      ===> _________ ===> __     ===> _________ ===> __
===> _________ ===> __      ===> _________ ===> __     ===> _________ ===> __
```

Figure 38. Report Header Layout Panel (ERB3RD5)

- Variable &ERBSAMPL that contains the number of samples replaces Z1.
- Variable &ERBSID that contains the session id replaces Z2 in the first header line of the report.
- &ERBRANGE that contains the range value replaces Z3 in the second header line.

Depending on your selection on panel ERB3RD3, you will continue as follows:
- If you specified TABULAR or BOTH for report mode, RMF displays the **Report Subheader Layout** panel ERB3RD6.
- If you specified GRAPHIC for report mode, RMF displays the **Graphic Parameter Definition** panel ERB3RDB, see “Graphic parameter definition panels (ERB3RDB, ERB3RDC, ERB3RDD)” on page 180.

Report subheader layout panels (ERB3RD6 and ERB3RD7)

The Report Subheader Layout panel (ERB3RD6) displays up to five subheader lines of an existing RMF report. (Subheader lines are any lines in an RMF report that appear between the two standard header lines and the column headings.) ERB3RD6 also lists the variables that are available for use in the subheader lines of the modified report.

At the bottom of ERB3RD6, you enter the subheader lines exactly as you want them to appear on your report. You can use the variables listed on the panel to appear in the subheadings of your report. Panel ERB3RD6 lists variables from header data table ERBHDRS3.

If a variable name is too long to appear in the header line, you can use a placeholder (&Z). After you press ENTER, you define these placeholders with variable names on the next panel.

Figure 39 is an example of a Report Subheader Layout panel ERB3RD6 that shows the subheadings of the SYSINFO report.

```
ERB3RD6                 RMF Report Format Definition
Command ===>

Report Name: SYSINFO    WLM Mode: GOAL       Section 4: Report Subheader Layout

Enter or change the report subheader lines.  To continue press ENTER.
To quit enter CANCEL. To go backwards press END.

You may intermix: text, variables, and variable placeholders (&Z).
If you specify variable placeholders (&Z) the next panel will ask you
to specify the variable name that is to replace each &Z.

The following variables are available for use in the subheader:
  &SYSPARVC    &SYSMODVC    &SYSMDLVC    &SYSTSVVC     &SYSIPVVC    &SYSPOLVC
  &SYSVEPVC    &SYSPRVVC    &SYSCUVVC    &SYSTSEVC     &SYSOPVVC    &SYSPADVC
  &SYSPRIVC    &SYSICVVC    &SYSLCPVC    &SYSAPIVC     &SYSPATVC    &SYSPRTVC
  &SYSAPTVC    &SYSAICVC    &SYSATCVC    &SYSLOAVG     &SYSTCTVC    &SYSUTCVC
Enter or change up to five report subheading lines:
&Z                    &Z   Model &Z           Appl%:     &Z   Policy: &Z
CPs Online:  &Z       Avg CPU Util%:  &Z      EAppl%:    &Z   Date:   &Z
AAPs Online: &Z    &Z                         Appl% AAP: &Z   Time:   &Z
IIPs Online: &Z                               Appl% IIP: &Z
```

Figure 39. Report Subheader Layout Panel (ERB3RD6)

In Figure 39, subheader lines appear at the bottom of the panel and 24 variable names from the SYSINFO report are available. You can modify these subheader lines and indicate where you want the available variables to appear in them.

Press ENTER to display the next panel ERB3RD7, the second Report Subheader Layout panel.

On this panel, you can specify variable names for any Z placeholders you have used. For a description of how to replace placeholders with variable names, see the Report Header Layout panel (Figure 38).

Figure 40 shows panel ERB3RD7 that defines placeholders used on the previous panel.

```
ERB3RD7                 RMF Report Format Definition
Command ===>

Report Name: SYSINFO    WLM Mode: GOAL       Section 4: Report Subheader Layout

The following report subheader lines have been specified:
Z1                     Z2   Model Z3          Appl%:     Z4   Policy: Z5
CPs Online:  Z6        Avg CPU Util%:  Z7     EAppl%:    Z8   Date:   Z9
AAPs Online: Z10   Z11                        Appl% AAP: Z12  Time:   Z13
IIPs Online: Z14                              Appl% IIP: Z15


Specify the placeholder (Z) number next to the variable name to replace each Z
above. To continue press ENTER. To go backwards press END.To quit enter CANCEL.

     &SYSPARVC ===> 1           &SYSMODVC ===> 2           &SYSMDLVC ===> 3
     &SYSTSVVC ===> 4           &SYSIPVVC ===> __          &SYSPOLVC ===> 5
     &SYSVEPVC ===> __          &SYSPRVVC ===> 6           &SYSCUVVC ===> 7
     &SYSTSEVC ===> 8           &SYSOPVVC ===> __          &SYSPADVC ===> 9
     &SYSPRIVC ===> 10          &SYSICVVC ===> __          &SYSLCPVC ===> 11
     &SYSAPIVC ===> 12          &SYSPATVC ===> 13          &SYSPRTVC ===> 14
     &SYSAPTVC ===> 15          &SYSAICVC ===> __          &SYSATCVC ===> __
     &SYSLOAVG ===> __          &SYSTCTVC ===> __          &SYSUTCVC ===> __
```

Figure 40. Report Subheader Layout Panel (fERB3RD7)

Report column layout panels (ERB3RD8 and ERB3RD9)

Press ENTER to display the next panel, ERB3RD8, the report subheader first **Report Column Layout** panel.

On this panel, you can modify report columns. You can enter up to three column header lines as you want them to appear in the report.

You can specify up to three model lines for your columns by using an attribute character followed by a variable name or placeholder (&Z). (See DATA ATTRIBUTE CHARACTERS described below.)

You can use the variable names listed at the bottom of the panel to appear in the columns of your report. This panel also allows you to specify a placeholder (&Z) for any variable name you want to use. (Panel ERB3RD8 lists variables from the data table of the RMF report you are modifying. All variables might not appear on the first page of the panel. Scroll through the panel and select the variable names you need. For information about RMF report data tables, see Chapter 8, “Monitor III data reporter tables,” on page 273.) You can define placeholders for variable names on the next panel.

Figure 41 is an example of ERB3RD8 that shows report column headings for the modified SYSINFO report with columns that contain data about TCB%, SRB%, and execution velocity. The details about delay percentages have been removed.

Column layout

```
ERB3RD8                 RMF Report Format Definition              Line 1 of 10
Command ===>                                                    Scroll ===> PAGE

Report Name: SYSINFO                                Section 5: Report Column Layout
WLM Mode:    GOAL

Enter or change the following information. To continue press ENTER.
To quit enter CANCEL. To go backwards press END.

DATA ATTRIBUTE CHARACTERS ===> _?|    Define meaning in attribute section
                                      of associated table display (ERB3SYS).

Enter or change up to three column header lines:
Group     T WFL --Users--  RESP TRANS  CPU   TCB   SRB  -AVG USG-   -Avg Del-
             %   TOT  ACT  Time  /SEC   %     %     %   PROC  DEV   PROC  DEV
_____________________________________________________________________________

Enter or change up to three model lines:
?Z        ?Z?Z  ?Z   ?Z    ?Z   ?Z    ?Z    ?Z    ?Z    ?Z   ?Z     ?Z   ?Z

_______________________________________________________________________________
_______________________________________________________________________________

The following variables are available for use in the model lines:
  SYSNAMVC     SYSTYPVC     SYSWFLVC     SYSTUSVC     SYSAUSVC     SYSTRSVC
  SYSAFCVC     SYSVECVC     SYSAUPVC     SYSAUDVC     SYSADPVC     SYSADDVC
  SYSADSVC     SYSADUVC     SYSADOVC     SYSADEVC     SYSADJVC     SYSADHVC
  SYSADXVC     SYSADNVC     SYSADMVC     SYSCPUVC     SYSSRBVC     SYSTCBVC
  SYSIFAVC     SYSCPVC      SYSIFCVC     SYSRSPVC     SYSVELVC     SYSUGMVC
  SYSUGPVC     SYSUGDVC     SYSWGDVC     SYSWGPVC     SYSDGMVC     SYSUJMVC
  SYSDJMVC     SYSDGEVC     SYSDGHVC     SYSDGDVC     SYSDGJVC     SYSDGOVC
  SYSDDSIP     SYSEAPVC     SYSLPIVC     SYSSUPVC     SYSSUCVC     SYSDTLLN
```

Figure 41. Report Column Layout Panel (ERB3RD8)

DATA ATTRIBUTE CHARACTERS

Specifies the ISPF characters used to indicate the start of a data field. Specify the data attribute characters before each variable name or placeholder (&Z) used in the model lines.

You must specify the name of a panel for the tabular version of a new or modified report. For RMF-supplied panels, the attribute characters appear as follows:

- a question mark (?) indicates that the output display characters appear unhighlighted (low intensity) in turquoise
- a slash (|) indicates that the output display characters appear highlighted (high intensity) in white
- a blank indicates that the input display characters appear unhighlighted (high intensity) in green

For user-defined panels, be sure that the data attribute characters match the characters in the attribute section of your ISPF display panel. See PANEL NAME on the report format information panel (ERB3RD3).

Press ENTER to display the next panel ERB3RD9, the second Report Column Layout panel.

On this panel, you can specify variable names for any Z placeholders you have used. The variable names available on the previous panel are listed at the bottom; you can add your own variable names in the spaces provided. If your variable names are not available when you invoke the report, blanks will appear instead of data. See the report header information panel (ERB3RD5) in Figure 38 for a description of how to replace placeholders with variable names.

If not all variable names appear on the first page of the panel, scroll through the remaining pages of the panel to see all available variable names.

Figure 42 is an example of Report Column Layout panel ERB3RD9 that defines placeholders used on the previous panel.

```
ERB3RD9                 RMF Report Format Definition              Line 1 of 21
Command ===>                                                      Scroll ===> PAGE

Report Name: SYSINFO                            Section 5: Report Column Layout
WLM Mode:    GOAL

The following report column header and model lines have been specified:
Group     T WFL --Users--  RESP TRANS  CPU   TCB   SRB  -AVG USG-   -Avg Del-
            %   TOT  ACT  Time  /SEC   %     %     %   PROC  DEV   PROC  DEV

Z1        Z2Z3  Z4   Z5    Z6   Z7    Z8    Z9    Z10   Z11  Z12    Z13  Z14



Specify the placeholder (Z) number next to the variable name to replace each Z
above. To continue press ENTER. To go backwards press END.To quit enter CANCEL.

     SYSNAMVC ===> 1            SYSTYPVC ===> 2            SYSWFLVC ===> 3
     SYSTUSVC ===> 4            SYSAUSVC ===> 5            SYSRSPVC ===> 6
     SYSTRSVC ===> 7            SYSCPUVC ===> 8            SYSTCBVC ===> 9
     SYSSRBVC ===> 10           SYSAUPVC ===> 11           SYSAUDVC ===> 12
     SYSADPVC ===> 13           SYSADDVC ===> 14           SYSAFCVC ===> __
     SYSVECVC ===> __           SYSADSVC ===> __           SYSADUVC ===> __
     SYSADOVC ===> __           SYSADEVC ===> __           SYSADJVC ===> __
     SYSADHVC ===> __           SYSADXVC ===> __           SYSADNVC ===> __
     SYSADMVC ===> __           SYSIFAVC ===> __           SYSCPVC  ===> __
     SYSIFCVC ===> __           SYSVELVC ===> __           SYSUGMVC ===> __
     SYSUGPVC ===> __           SYSUGDVC ===> __           SYSWGDVC ===> __
     SYSWGPVC ===> __           SYSDGMVC ===> __           SYSUJMVC ===> __
     SYSDJMVC ===> __           SYSDGEVC ===> __           SYSDGHVC ===> __
     SYSDGDVC ===> __           SYSDGJVC ===> __           SYSDGOVC ===> __
     SYSDGPVC ===> __           SYSDGSVC ===> __           SYSDGUVC ===> __
```

Figure 42. Report Column Layout Panel (ERB3RD9)

Command line layout panel (ERB3RDA)

Press ENTER to display the next panel ERB3RDA, the **Command Line Layout** panel.

On this panel, you can specify the format of the command line and scroll line as you want them to appear on the hardcopy of the tabular report. You must also define the command line and scroll line on the display panel of the tabular report.

Figure 43 is an example of Command Line Layout panel ERB3RDA.

```
ERB3RDA                 RMF Report Format Definition
Command ===>

Report Name: SYSCPU     WLM Mode: GOAL       Section 6: Command Line Layout


Enter or change the following information.  To continue press ENTER.
To quit enter CANCEL. To go backwards press END.

You may intermix: text, variables, and variable placeholders (&Z).

The following variables are available for use in the command line:
  &ZCMD          &AMT

Enter or change the command line:

Command ===>

Specify a variable name in each of the entry fields to replace each Z above.

Z1 ===>
Z2 ===>
Z3 ===>
```

Figure 43. Command Line Layout Panel (ERB3RDA)

Graphic parameter definition panels (ERB3RDB, ERB3RDC, ERB3RDD)

If you specified BOTH or GRAPHIC for report mode on ERB3RD3, RMF displays the first **Graphic Parameter Definition** panel, ERB3RDB.

On this panel, you can specify general information about the graphic version of the report.

Note: If you specified TABULAR for report mode on the report format information panel (ERB3RD1) or used DI or WFEX as a prototype, the report format definition utility displays panel ERB3RDF. This panel allows you to save your changes and view the tabular report you have created or cancel your changes. See "Saving or cancelling changes on panel ERB3RDF" on page 184.

Figure 44 is an example of the Graphic Parameter Definition panel ERB3RDB that specifies general information for the graphic version of the SYSINFO report:

```
ERB3RDB                 RMF Report Format Definition
Command ===>

Report Name: SYSCPU                     Section 7: Graphic Parameter Definition
WLM Mode:    GOAL
             Definitions on this panel are independent of WLM mode.

Enter the following information.  To continue press ENTER.
To quit enter CANCEL. To go backwards press END.


                              GENERAL INFORMATION



NAME FOR HELP PANEL ===> ERBGSYS0     Name of HELP PANEL, if any

TITLE FOR Y-AXIS    ===> Average Number of Active Users

MINIMUM AXIS RANGE  ===> 1            Axis will contain at least this
                                      number of data points
SELECTION RULE      ===> 1            Specify 0, 1, 2 or 3
```

Figure 44. Graphic Parameter Definition Panel (ERB3RDB)

The fields and their meanings follow:

NAME FOR HELP PANEL
Specifies the name of the help panel that you provide for the graphic report. The field is optional.

TITLE FOR Y-AXIS
Specifies a line of text (maximum of 50 characters) to appear as a label for the bar graph in the graphic version of the report. Sample lines that appear in the graphic parameter table (ERBPTGS3) are:

- Percentage of Each User's Time
- Percentage of the User's Time
- Average Number of Active Users

For an example of the graphic parameter table (ERBPTGS3), see Chapter 8, “Monitor III data reporter tables,” on page 273.

MINIMUM AXIS RANGE
Specifies the length of the bar graph depending on the text specified in TITLE FOR Y-AXIS as follows. For each line of text listed in the previous example, the minimum axis range is as follows:

- 100 for “Percent of Each User's Time”
- 100 for “Percent of the User's Time”
- 1 for “Average Number of Active Users”

If the length of the largest bar in the report exceeds the value you specify, RMF uses the length of the largest bar.

For an example of the graphic parameter table (ERBPTGS3), see Chapter 8, “Monitor III data reporter tables,” on page 273.

SELECTION RULE
Specifies how the lines of the tabular report appear as bar graphs on the graphic version of the report. You can select one of the following values:

- 0 - One bar corresponds to one line of the RMF tabular report

- 1 - One bar corresponds to one line of the RMF tabular report with sequence number 1 (for example, DEV, HSM, JES, STOR, PROC, DELAY, SYSINFO, and ENQ)
- 2 - One bar corresponds to the summary of logical lines of the report (for example, ENQR, DEVR reports)
- 3 - Two bar types can result from all logical lines of a logical block in the RMF tabular report (for example, STORR report) as follows:
  - Bar type 1 corresponds to a line of the tabular report with sequence number 1
  - Bar type 2 corresponds to each additional line of the logical block for a tabular report with a sequence number greater than 1

  For an example of the graphic parameter table (ERBPTGS3), see Chapter 8, "Monitor III data reporter tables," on page 273. For a description of logical line number and sequence number, see the panel field description for ERB3RD3 (Figure 36).

Press ENTER to display the next panel, ERB3RDC, the second **Graphic Parameter Definition** panel.

On this panel, you can specify labels for the graphic bars in the report. You can specify variable names for bar type 1 labels and bar type 2 labels.

Figure 45 is an example of the Graphic Parameter Definition panel ERB3RDC.

```
ERB3RDC                 RMF Report Format Definition
Command ===>

Report Name: SYSCPU                     Section 7: Graphic Parameter Definition
WLM Mode:    GOAL
             Definitions on this panel are independent of WLM mode.

Enter the following information.  To continue press ENTER.
To quit enter CANCEL. To go backwards press END.

                        LABEL INFORMATION FOR BAR TYPE I

PRIMARY LABEL       ===> SYSNAMVC  Variable name containing label
SECONDARY LABEL     ===> ________  Variable name containing label
PRIMARY COMPOSITE   ===> _____     Prefix of label
SECONDARY COMPOSITE ===> _____     Prefix of label

                       LABEL INFORMATION FOR BAR TYPE II

PRIMARY LABEL       ===> ________  Variable name containing label
SECONDARY LABEL     ===> ________  Variable name containing label
PRIMARY COMPOSITE   ===> _____     Prefix of label
SECONDARY COMPOSITE ===> _____     Prefix of label
```

Figure 45. Graphic Parameter Definition Panel (ERB3RDC)

The panel fields and their meanings are as follows:

PRIMARY LABEL/SECONDARY LABEL
: Specifies an 8 character variable name for a data value in the graphic version of the report. You can use the variable names that appear in the ISPF data table of the corresponding tabular report.

  For example, in Figure 45, the Primary label will appear as average number of active users (SYSNAMVC) on the graphic version of the SYSCPU report.

See Chapter 8, “Monitor III data reporter tables,” on page 273 for examples of the Graphic Parameter table (ERBPTGS3) and the RMF Report Data tables.

PRIMARY COMPOSITE/SECONDARY COMPOSITE
Specifies up to 5 characters of text as a prefix to the variable label specified in PRIMARY/SECONDARY LABEL. In Figure 45, no composite labels appear in the SYSCPU report. You can specify a prefix to appear in the graphic version of the report. The prefix is concatenated to the rightmost contents of the report table variable specified in PRIMARY/SECONDARY label.

See Chapter 8, “Monitor III data reporter tables,” on page 273 for examples of the Graphic Parameter table (ERBPTGS3) and the RMF Report Data tables.

BAR TYPE refers to the number of bars used in the report depending on the logical line and sequence numbers.

See LOGICAL LINE/SEQUENCE NUMBER in the field descriptions for ERB3RD3 (Figure 36).

Press ENTER to display the next panel, ERB3RDD, the third **Graphic Parameter Definition** panel.

On this panel, you can specify data columns that you want to appear in the graphic version of the report.

Figure 46 is an example of the Graphic Parameter Definition panel ERB3RDD.

```
ERB3RDD                 RMF Report Format Definition
Command ===>

Report Name: SYSCPU                         Section 7: Graphic Parameter Definition
WLM Mode:    GOAL
             Definitions on this panel are independent of WLM mode.

Enter the following information. To continue press ENTER.
To quit enter CANCEL. To go backwards press END.

                   COLUMN SPECIFICATION FOR GRAPHIC BAR TYPES

         NAME              LEGEND ID           TRANS ID           BAR TYPE ID

 1. ===> SYSADPVC       ===> 14             ===> 0              ===> 1
 2. ===> SYSADDVC       ===> 08             ===> 0              ===> 1
 3. ===> SYSADSVC       ===> 15             ===> 0              ===> 1
 4. ===> SYSADUVC       ===> 28             ===> 0              ===> 1
 5. ===> SYSADOVC       ===> 29             ===> 0              ===> 1
 6. ===> SYSADEVC       ===> 09             ===> 0              ===> 1
 7. ===> SYSAUPVC       ===> 19             ===> 0              ===> 1
 8. ===> SYSAUDVC       ===> 18             ===> 0              ===> 1
 9. ===> ________       ===> __             ===> _              ===> _
10. ===> ________       ===> __             ===> _              ===> _
```

Figure 46. Graphic Parameter Definition Panel (ERB3RDD)

The panel fields and their meanings are as follows:

NAME
Specifies an 8 character variable name for a data value from the corresponding tabular report. This value will appear as a bar column in the graphic version of the report. The bar column can be a single bar (bar type 1) or a stacked bar

(bar type 2) depending on what you specify for BARTYPE ID. See Chapter 8, “Monitor III data reporter tables,” on page 273 for examples of RMF report data tables.

LEGEND ID
Specifies a number that corresponds to the color, pattern and the text of the graphic chart legend. Variables specified for NAME will appear in the color specified for LEGEND ID. You can specify a decimal value from 04 to 27; the numbers must match the color ID entries on the Color Graphic Option panels.

TRANS ID
Specifies a number that controls how the values for the variable in NAME are scaled on the bar graph in the graphic version of the report.
- 0 - value appears as is; no division is performed
- n - value is divided by 10^n where **n** equals an integer from 1 to 9.

See Chapter 8, “Monitor III data reporter tables,” on page 273 for examples of RMF report data tables.

BARTYPE ID
Specifies a value that indicates where the data value for the variable in NAME appears for bar types in the graphic version of the report:
- 0 - indicates the value appears in both bar types
- 1 - indicates the value occurs in bar type 1
- 2 - indicates the value occurs in bar type 2

If you specified label information for only bar type 1 on the report parameter definition panel (ERB3RDC), you must specify bar type 1.

Saving or cancelling changes on panel ERB3RDF

Once you have created or modified a report using the report format definition utility panels, RMF displays panel ERB3RDF, which allows you to confirm or cancel your changes.

```
ERB3RDF                 RMF Report Format Definition
Command ===>


This is a confirmation/cancellation panel for report: SYSCPU
                               related with WLM mode: GOAL



The following actions are allowed:


       Type  SAVE   command to save report

       Type  CANCEL command to cancel processing

       Press END    key to go one step backwards

       Press ENTER  key to see the sample report
```

Figure 47. Configuration/Cancellation Panel (ERB3RDF)

You can get a report with sample data just to verify the correct layout of the report. In this example, some values are not displayed because they are not part of the sample data.

```
ERB4CPU                 RMF V1R13  CPU Information               Line 1 of 20
Command ===>                                                 Scroll ===> PAGE

Samples:     60  System: SYSE  Date: 06/08/2011  Time: 08.59.00  Range:     60Sec

Partition:    SYSE    2084  Model 314         Appl% :     22  Policy: STANDARD
CPs Online:   4.0     Avg CPU Util%:   24     EAppl%:     22  Date:   05/12/05
AAPs Online:  1.0     Avg MVS Util%:   26     Appl% AAP:  19  Time:   12.20.27
IIPs Online:    -                             Appl% IIP:   -

Group     T WFL --Users--  RESP TRANS  CPU   TCB   SRB  -AVG USG-   -Avg Del-
              %  TOT  ACT  Time  /SEC   %     %     %   PROC  DEV   PROC  DEV

*SYSTEM      98  167    2        0.12  21.6  21.5   0.2  1.6  0.1    0.0  0.0
*TSO               3    0        0.12   0.0   0.0   0.0  0.0  0.0    0.0  0.0
*BATCH       99    3    2        0.00  20.5  20.5   0.0  1.5  0.0    0.0  0.0
*STC         91  158    0        0.00   1.2   1.0   0.2  0.1  0.1    0.0  0.0
*ASCH              0    0        0.00   0.0   0.0   0.0  0.0  0.0    0.0  0.0
*OMVS              3    0        0.00   0.0   0.0   0.0  0.0  0.0    0.0  0.0
*ENCLAVE           0  N/A         N/A   N/A   N/A   N/A  0.0  N/A    0.0  N/A
BATCH     W  99    4    2  .000  0.00  20.5  20.5   0.0  1.5  0.0    0.0  0.0
BATCHLOW  S  99    3    2  .000  0.00  20.5  20.5   0.0  1.5  0.0    0.0  0.0
OMVSKERN  S        1    0  .000  0.00   0.0   0.0   0.0  0.0  0.0    0.0  0.0
OMVS      W        2    0  .000  0.00   0.0   0.0   0.0  0.0  0.0    0.0  0.0
OE        S        2    0  .000  0.00   0.0   0.0   0.0  0.0  0.0    0.0  0.0
STC       W 100   26    0  .000  0.00   0.1   0.1   0.0  0.0  0.0    0.0  0.0
STCDEF    S 100   26    0  .000  0.00   0.1   0.1   0.0  0.0  0.0    0.0  0.0
SYSTEM    W  90  132    0  .000  0.00   1.0   0.9   0.2  0.1  0.1    0.0  0.0
SYSSTC    S 100  111    0  .000  0.00   0.3   0.3   0.0  0.0  0.0    0.0  0.0
SYSTEM    S  88   21    0  .000  0.00   0.7   0.6   0.2  0.1  0.1    0.0  0.0
TSO       W        3    0  .010  0.17   0.0   0.0   0.0  0.0  0.0    0.0  0.0
TSODEF    S        3    0  .010  0.17   0.0   0.0   0.0  0.0  0.0    0.0  0.0
BCP       R  99    3    2  .000  0.00  20.5  20.5   0.0  1.5  0.0    0.0  0.0
```

Figure 48. Initial Version of the SYSCPU Report

The report shows that adjustments for some columns are necessary. You can do this either by stepping back to panel ERB3RD8 before you save the report or by modifying the stored report.

Enter SAVE to save the report or CANCEL to cancel your changes and return to the report definition initialization panel (ERB3RD1). If you save the report, RMF redisplays panel ERB3RD1 with a message that tells you the report has been modified or created. To exit the sample report panel and return to panel ERB3RDF, press END.

Deleting a user-defined report

If you specify DELETE for a report on the report format definition panel, RMF displays panel ERB3RDE. To confirm the deletion of the report, press ENTER and the report is deleted. To cancel the deletion, type CANCEL and press ENTER. RMF returns you to ERB3RD1.

Note: You can only delete a user-defined report. RMF does not allow you to delete an existing RMF report.

Ending the report utility

You can end the report format definition utility session by pressing END (PF3) on the report format definition panel (ERB3RD1) or by specifying CANCEL on any panel.

Implementing the report

To make the new SYSCPU report available, it needs to be integrated in a Monitor III selection panel. As defined initially, the report shall be added to the User Selection menu ERB3USR. You can do this by these modifications to the definition of the panel:

```
)attr default(!+_)
/*********************************************************************
/*          PANEL NAME: ERB3USR
/*

...

)body expand("") cmd(zcmd)
+          !         RMF User-written Report Selection Menu      " "
!Selection ===>_ZCMD " " +
+
<Enter selection number or command for desired report.
+
+
!  1<MSI           +Migration SYSINFO including Execution Velocity
!  2<DSD           +Detailed Storage Delays
!  3<RG            +Resource Group Data
!  4<SYSCPU        +Modified SYSINFO including CPU details
+

...
```

Figure 49. Modifications in User Selection Menu Definition (ERB3USR) - Part 1

```
/* translate subsystem selections ********************************* */
   &erbcmdc = trans(&erbcmdc
                  1,'MSI'
                  2,'DSD'
                  3,'RG'
                  4,'SYSCPU'
                 ST,'SYSTREND'
                 DA,'DEVN'
                 DT,'DEVT'
                  *,*)
/* Checks if command input is a valid RMF command.                  */
   ver(&erbcmdc,LIST, CANCEL, FIND, GRAPHIC, ICU, HARDCOPY, RESET,
       RFIND, TABULAR, TOGGLE,
       MSI, DSD, RG,  SYSCPU,
       DEVN, DEVT, SYSTREND,
       MSG=ERB562I)
/* Checks if command input is a valid on this screen.               */
   ver(&erbcmdc,LIST,;
       MSI, DSD, RG,  SYSCPU,
       DEVN, DEVT, SYSTREND,
       MSG=ERB573I)

...

/* selects action according to entered input ********************** */
   &zsel = trans(&erbcmdc
             MSI,'PGM(ERB3RDPC) PARM(MSI)'
             DSD,'PGM(ERB3RDPC) PARM(DSD)'
              RG,'PGM(ERB3RDPC) PARM(RG)'
          SYSCPU,'PGM(ERB3RDPC) PARM(SYSCPU)'
            DEVN,'PGM(ERB3RDPC) PARM(DEVN)'
            DEVT,'PGM(ERB3RDPC) PARM(DEVT)'
        SYSTREND,'PGM(ERB3RDPC) PARM(SYSTREND)'
               *,' ')
)END
```

Figure 50. Modifications in User Selection Menu Definition (ERB3USR) - Part 2

If you call the User Selection menu, you now get this new version:

```
ERB3USR             RMF User-written Report Selection Menu
Selection ===>

Enter selection number or command for desired report.


  1 MSI             Migration SYSINFO including Execution Velocity
  2 DSD             Detailed Storage Delays
  3 RG              Resource Group Data
  4 SYSCPU          Modified SYSINFO including CPU details

Device Reports
 DA DEVN            Device Activity
 DT DEVT            Device Trend
                    Device   => _______

System Reports
 ST SYSTREND        System and Workload Trend
                    Workload => ________
```

Figure 51. Modified User Selection Menu (ERB3USR)

Special considerations for modifying reports

If you want to add or delete lines in an existing RMF report or sort lines of a report without modifying the report heading, consider the following when you use the report format definition utility:

- Each RMF report data table (PHDRTAB1 in the phase driver table) contains the ISPF key type variables for the logical line number and line sequence number for the report. Each data table lists the logical lines and the sequence number(s) for logical lines of data in the report in ascending order. Sequence numbers for each logical line begin with 1. When you add, delete, or sort lines of an RMF report, be sure that the output table of your report (PHASE 1 or 2 TABLE on phase driver information panel ERB3RD2) arranges logical line and sequence numbers in ascending order.
- If you delete a line of a report with sequence number 1, you must also delete the logical line number of the report from the data table.
- If you want to rearrange the lines of an RMF report, you can use the ISPF service TBSORT as part of the CLIST you specify for phase 2. You can specify the CLIST with TBSORT on the phase driver information panel (ERB3RD2) as follows:

  ```
  CMD(mysort)
  ```

  where “mysort” is the name of your CLIST.

Installing your own phases

When you select a report during a reporter session, RMF uses ISPF SELECT services to generate report data tables and display the reports. You can supply your own routines for any of the 4 phases to produce user-defined reports. See “Data reporter phases” on page 163 for a description of the phases RMF invokes.

The following ISPF shared variables are available during all phases. They can be updated in Phase 1 by the Data Retrieval Service.

ERBDATE,ERBTIME
The ISPF shared pool variables that contain the beginning date and time of the display data.

ERBRANGE
The ISPF shared pool variable that contains the range time of the display data. The beginning date/time plus the range time of the display data equals the ending date/time.

ERBSID
The ISPF shared pool variable that contains the id of the system on which the data was collected.

ERBSAMPL
The ISPF shared pool variable that contains the number of data samples for the time range.

ERBRMFVD
The ISPF shared pool variable that contains the RMF version number of the data gatherer which collected the data. The format is **RMF VvRr** (that is, RMF V1R13).

ERBDTBEG
The ISPF shared pool variable which contains the beginning date/time value for the requested time range. The format is **MMDDYYYYHHMMSS** (that is, 09252011183050 represents September 25, 2011 at 18:30:50).

ERBDTEND
The ISPF shared pool variable which contains the ending date/time value for the requested time range. The format is **MMDDYYYYHHMMSS** (that is, 08272011173010 represents August 27, 2011 at 17:30:10).

ERBMNTIM The ISPF shared pool variable which contains the Monitor III data gatherer MINTIME option value that was in effect when the data was gathered. The data is in external decimal format.

Phase 1

If you want to use your own program for phase 1, you must ensure that the ISPF shared pool variable PHDRPH1 contains the name of your program or CLIST. This variable appears in the phase driver table (ERBPHDS3) as an ISPF SELECT string. For RMF reports, the PARM value of the string matches the name of the RMF report command. You can use the report definition format utility to specify your own PHASE 1 SELECT STRING. See Chapter 8, “Monitor III data reporter tables,” on page 273 for an example of the phase driver table (ERBPHDS3) entries and how they are specified.

If you want to change the time range from which your data is collected, you can invoke the Data Retrieval Service (ERB3RDRS) module from your phase 1 program. See “Data retrieval service (ERB3RDRS)” on page 192 for information about how to invoke the Data Retrieval Service.

The following ISPF shared variables contain information that RMF uses to generate a report during phase 1:

ERBREPC The ISPF shared pool variable that contains the current command or report selection. RMF uses this variable as a key to ERBPHDT3, the phase driver table. This table has an entry (in the table field PHREPNA) for each RMF command or report selection. RMF retrieves the necessary information to generate the report during phase 1 from ERBPHDT3 (a copy of ERBPHDS3).

ERBCMDC,ERBPARC
The ISPF shared pool variables that contain the current command (ERBCMDC) and any command parameters (ERBPARC).

ERBSSHG3 The ISPF shared pool variable that contains the address of the set-of-samples header (SSHG3). This control block contains the addresses of the sample data that correspond to the time and range values specified during the Monitor III data gatherer session or as indicated by the Data Retrieval Service. (See Figure 30 for an example of ERBSSHG3.)

ERBSUPP The ISPF shared pool variable that contains the number of the subpool that non RMF functions must use for GETMAINS.

During phase 1, the phase driver module (ERB3RPH1) uses the information in the report row entry of ISPF table ERBPHDT3 (a copy of ERBPHDS3) to produce the RMF report. If you supplied your own program or CLIST for phase 1, that routine gets control.

Upon completion, phase 1 must provide the following output:

ERBREPC The ISPF shared pool variable that should be restored to its value at entry to phase 1.

PHDRET1 The ISPF shared pool variable that should contain your return code from the program or CLIST used in phase 1.

For RMF supplied reports, ERB3RPH1 creates the report in phase 1 and returns one of the following return codes:

0 ISPF table successfully generated for the report

4 ISPF table generated for the report has some data, but errors have occurred

8 ISPF table generated for the report has no data, and an error has occurred

For your own routine, you might want to use the same return codes.

PHDRTAB1 The ISPF shared pool variable that contains the name of the ISPF data table generated in phase 1. If you omit phase 2, you must ensure that PHDRTAB2 contains the same name as PHDRTAB1. See phases 2 and 3 described later.

You can define your own ISPF shared pool variables to contain information that you want to include for phase 1. To ensure that no interference with RMF created variables occurs, the first three characters of user-defined variables should appear as follows:

```
USR
```

Phase 2

For phase 2, you supply a program or CLIST to modify the ISPF table created for the report in phase 1.

The following ISPF shared variables contain information for phase 2:

ERBREPC The ISPF shared pool variable that should contain the current command or report selection.

ERBCMDC,ERBPARC
The ISPF shared pool variables that should contain the current command (ERBCMDC) and any command parameters (ERBPARC).

PHDRET1 The ISPF shared pool variable that should contain your return code from the program or CLIST used in phase 1.

For RMF supplied reports, ERB3RPH1 creates the report in phase 1 and returns one of the following return codes:

0 ISPF table successfully generated for the report

4 ISPF table generated for the report has some data, but errors have occurred

8 ISPF table generated for the report has no data, and an error has occurred

For your own routine, you might want to use the same return codes.

PHDRTAB1 The ISPF shared pool variable that contains the name of the ISPF data table generated in phase 1.

Upon completion, phase 2 must provide the following output:

ERBREPC The ISPF shared pool variable that should be restored to its value at entry to phase 1.

PHDRET2 The ISPF shared pool variable that should contain the return codes from the RMF program or CLIST used to create the report in phase 2.

PHDRTAB2 The ISPF shared pool variable that should contain the name of the ISPF data table generated in phase 2. You can use the same table name as for PHDRTAB1.

Phase 3

For phase 3, RMF formats the ISPF table generated in phase 1 or 2 and displays the report. To format the ISPF report data tables, RMF uses the tabular report format table (ERBFMTS3), the RMF header table (ERBHDRS3), and the graphic parameter table (ERBPTGS3). The RMF display phase module (ERB3RDSP) displays the report by means of the ISPF TBDISPL service.

The following ISPF shared variables contain information that you can use to format and display a report during phase 3:

ERBREPC
The ISPF shared pool variable that contains the current command or report selection. The variable is a key to obtain formatting information for the tabular report in the report format table (ERBFMTS3) or the graphic report in the graphic parameter table (ERBPTGS3). For examples of these tables, see Appendix B.

ERBCMDC,ERBPARC
The ISPF shared pool variables that contain the current command (ERBCMDC) and any command parameters (ERBPARC).

PHDRET1,PHDRET2
The ISPF shared pool variables that should contain return codes from phase 1 and 2.

PHDRTAB2
The ISPF shared pool variable that should contain the name of the ISPF data table generated in phase 1 and/or phase 2.

SESRPFU3
The ISPF shared pool variable that contains the report mode (TABULAR or GRAPHIC).

RMF uses module ERB3RDSP to display the reports. The module dynamically constructs a panel from information in the format tables. It builds header and model lines and constructs the graphic area within the panel and uses the ISPF data table whose name appears in the ISPF shared pool variable PHDRTAB2.

Upon completion, phase 3 must provide the following output:

ERBREPC
The ISPF shared pool variable that should be restored to its value at entry to phase 1.

PHDRET3
The ISPF shared pool variable that should contain the return code from the program or CLIST used to format and display the report.

If you decide to replace the RMF module ERB3RDSP with your own routine, you must consider the following:

- To obtain a display of your reports in GO mode, you must invoke the ISPF service CONTROL LOCK before the ISPF service TBDISPL is performed.
- Your module must handle all ISPF PASSTHRU commands.

Phase 4

For phase 4, you provide a program that can perform cleanup services for resources you might have used during previous phases. For example, if you have used ISPF TBCREATE with the WRITE SHARE option to create an ISPF table, you can use ISPF TBEND to delete the table during phase 4. Or use TBEND to save and then delete the table. See the ISPF publications that describe these services for more information.

The following ISPF shared variables contain information that you can use to format and display a report during phase 4:

ERBREPC
The ISPF shared pool variable that contains the current command or report selection.

ERBCMDC,ERBPARC
The ISPF shared pool variables that contain the current command (ERBCMDC) and any command parameters (ERBPARC).

ERBSUPP
The ISPF shared pool variable that contains the number of the subpool used for GETMAINS.

PHDRET1,PHDRET2,PHDRET3
The ISPF shared pool variables that should contain return codes from phase 1, 2, and 3.

Upon completion, phase 4 must provide the following output:

ERBREPC
The ISPF shared pool variable that should be restored to its value at entry to phase 1.

PHDRET4
The ISPF shared pool variables that should contain return codes from phase 4.

Data retrieval service (ERB3RDRS)

The Data Retrieval Service (ERB3RDRS) module provides flexibility for user exits to change the time range from which data is collected. The module is called from phase 1 of your user exit. This service can be invoked by either calling it,

Example

```
CALL ERB3RDRS (PARMAREA)
```

or by using the ISPF SELECT service.

Example

```
ISPEXEC SELECT PGM(ERB3RDRS) PARM(PARMAREA)
```

To use this service, the caller must invoke the module ERB3RDRS with the registers and parameter area described in “Parameter area contents” on page 193.

Programming considerations

Do not link the module ERB3RDRS to your application program. Assembler programs must use LOAD or LINK macros to access the module; PL/I programs must use FETCH/RELEASE; and C programs must use the builtin function FETCH.

The caller must be in 31-bit addressing mode and can run unauthorized.

Function codes

The function code specifies the time range to be used by the Data Retrieval Service:

1 Most recent number of MINTIMEs (as in GO mode)

2 Retrieve data from the range determined by BEG Date and Time and END Date and Time (similar to the BREF command with parameters DATE=,TIME=, and RANGE=)

3 Retrieve data from the range determined by using END Date and Time as end time, and going backward in time using the current RANGE (similar to the BREF command without parameters)

4 Retrieve data from the range determined by BEG Date and Time as begin time, and going forward in time using the current RANGE (similar to the FREF command without parameters)

Registers at entry

The contents of the registers on entry to this service are:

Register	Contents
0	Not used
1	Parameter list address
2-12	Not used
13	Standard save area address
14	Return address
15	Entry point address of ERB3RDRS

Parameter area contents

The parameter area passed by the caller to the RMF Data Retrieval Service is a single character string, preceded by a halfword containing the length of the parameter area in binary. The parameter area is as follows:

First word Bytes 0 to 3: function code

Second word Bytes 4 to 7: number of MINTIMEs (this is used only with function code 1)

Character string
Bytes 8 to 21: begin date and time of the requested time range in character format of MMDDYYYYHHMMSS.

Character string
Bytes 22 to 35: end date and time of the requested time range in character format of MMDDYYYYHHMMSS.

Output

The Data Retrieval Service module updates the following shared pool variables:

ERBSSHG3 The ISPF shared pool variable that contains the beginning address of the common set of samples. If no data could be retrieved, this variable is set to hexadecimal zero.

ERBDTBEG The ISPF shared pool variable that contains the beginning date/time value of the retrieved range.

ERBDTEND The ISPF shared pool variable that contains the ending date/time value of the retrieved range.

ERBMNTIM The ISPF shared pool variable that contains the Monitor III data gatherer MINTIME option value in external format.

Return codes

Upon return from this service, register 15 provides the return code and reason code as listed in Table 15:
- Bytes 0 and 1 are not used (x'0000')
- Byte 2 contains reason code

- Byte 3 contains return code

Table 15. Return and Reason Codes for the Data Retrieval Service (ERB3RDRS)

Return Code (Decimal)	Reason Code (Decimal)	Description
0	0	Data returned with no errors.
4	4	Data might be inconsistent due to a SET IPS change detected within the specified range. This is valid for data being gathered with RMF Version 4.
8		Data only partially returned.
	8	Partial data returned. Message ERB589I displayed.
	9	VSAM retrieval error occurred. Partial data returned. Message ERB589I displayed.
	13	The WLM service policy has changed, or the IPS values have been modified. This is valid for data being gathered with RMF Version 5 and above.
	14	The RMF cycle time has changed.
	15	IPL detected.
12		No data returned.
	8	No data returned. Message ERB587I displayed.
	9	VSAM retrieval error occurred. No data returned. Message ERB587I displayed.
	14	Cycle time changed. Message ERB559I displayed.
	15	IPL detected. Message ERB558I displayed.
	16	No data available. Message ERB591I displayed
	17	Data gatherer is not active. Message ERB565I displayed.
	18	Preallocated data sets are unusable. Message ERB583I displayed.
	19	Preallocated data sets found to be unusable during data retrieval. Message ERB583I displayed.
	20	Too many reporters tried to access the in-storage buffer. Message ERB564I displayed.
	21	Retrieval from in-storage buffer failed. Message ERB564I displayed.
	22	No data is in the in-storage buffer. Message ERB591I displayed.
	23	Insufficient storage to copy data from the in-storage buffer. Message ERB564I displayed.
16	0	Incorrect function code.

Note: The RMF Monitor III standard reports provide information on the same time range as was requested in the last use of the Data Retrieval Service.

End of Programming Interface information

TSO/E user authorization

Programming Interface information

TSO/E must be installed on your system to use the following commands.

All the data collected and reported by RMF during a Monitor III display session is obtained from commonly addressable storage that is not fetch protected. However, if your installation wants to limit the use of the command that starts an RMF Monitor III session under TSO/E, one method available is to replace the RMF control section with your own module. For Monitor III you replace the control section ERB3SOCK. Your routine will then be invoked as part of the RMF response to the RMF command.

ERB3SOCK (Monitor III) runs in problem state with a key of 8. When this control section gets control, register 1 points to a two-word address list. The first address points to the seven-byte user ID of the user who has entered the RMF command. The second word points to the PSCB. Figure 52 illustrates the input parameter structure.

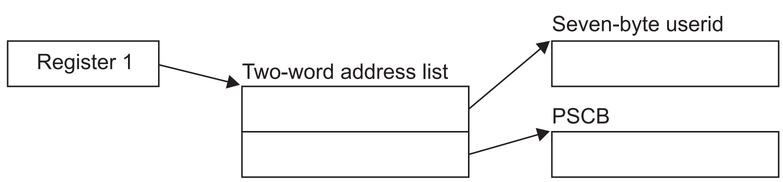


Figure 52. ERB3SOCK Input Parameter Structure

The module that you code to replace ERB3SOCK must be reenterable. It receives control by a BALR instruction and must save the registers when it receives control and restore the registers when it returns control. Register 13 contains the address of the register save area; register 14 contains the return address; and register 15 contains the entry address.

The processing your module performs depends on the method you choose to validate the user. Possible methods include issuing a RACHECK, prompting the user for a password, or checking the userid against a list of valid userids. Information on the TSO/E services available to perform these functions, such as TGET or TPUT, can be found in *z/OS TSO/E Programming Services*.

You can also use the PSCB bits defined for user use. This field (PSCBATR2 in the PSCB) comes from the UADS and can be updated by the USERDATA keyword of the ADD and CHANGE subcommands of the ACCOUNT command. See *z/OS TSO/E System Programming Command Reference* for more information on these commands.

When your routine has completed its processing, set a return code of 0 in register 15 to indicate to RMF that the user is authorized to enter RMF. Set a return code of 4 in register 15 to indicate to RMF that the user is not authorized to enter RMF. In response to this return code, RMF will display a message at the display station. No session will be started. After setting the appropriate return code, return control by branching on the contents of register 14.

For the Monitor III TSO/E session the user authorization exit routine (ERB3SOCK) is part of the RMF load module that contains the RMF command. This module resides in SYS1.SERBLINK as load module RMF; its entry point is ERB3RTSO. Before your authorization routine can run, you must link edit it with RMF; the control statements required are:

```
(ERB3SOCK object deck)
INCLUDE ddname(RMF)
ENTRY ERB3RTSO
NAME RMF(R)
```

End of Programming Interface information

Chapter 7. Using Monitor III VSAM data set support

This topic provides the following information:

- It describes the data set structure and content for the Monitor III data set support function
- It lists the record fields and table entries associated with data set support

See the *z/OS RMF User's Guide* for more information about data set support and recording.

Data set record structure

Programming Interface information

If no specific limitation is stated, then all fields in the records, including those indicated as `RESERVED FOR USER`, but **excluding** all others indicated as `RESERVED` are part of the programming interface.

With the data set support function, RMF uses VSAM relative record data sets (RRDS) to record measurement information during a Monitor III gatherer session.

During data set recording RMF collects measurement data in the form of one set of samples for each MINTIME and records the samples on the VSAM data sets. Before storing the data, RMF compresses the data one MINTIME at a time. The data is stored in compressed format except for the Data Set Header and Index Table (ERBDSIG3) and the MINTIME Set of Samples Header Table (ERBSSHG3). The description of the data tables are valid only after the RMF decompression interface (ERB3RDEC) is used to decompress the data one MINTIME at time. The RMF Monitor III reporter will decompress the data after retrieving it from the VSAM data sets. To directly access the VSAM data sets and process them without the use of the Monitor III reporter, use the service module, ERB3RDEC. See "Data set decompression" on page 198 for more information.

RMF data can be accessed directly by relative record number or by sequential records. Each data set is a string of fixed-length records, and each record is identified by a relative record number. Because RMF treats the data it records on the data set as a linear data set, it writes the logical records as a contiguous stream of sampled data with little dependency on the record size. To allow retrieval of the data, an index relates the time stamp of every MINTIME set of samples with the offset of the set of samples within the data set and its length; therefore, you can determine the relative record number of any given set of samples within a data set by dividing the offset and the length of the set of samples by the record length, which is 32,752 bytes. (Note: VSAM does not maintain the index.)

The first record on every VSAM data set contains the data set header. It is followed by the index information (see "ERBDSIG3 - Data set header and index" on page 229). RMF builds one index entry for each MINTIME set of samples in the data set. When no more entries can fit into the index, RMF closes the data set. The records in the data set following the index information contain the measurements of each MINTIME set of samples (see "ERBSSHG3 - MINTIME set of samples header" on page 263). RMF stores data on the data set as follows:

- contiguously arranges MINTIME sets of samples in chronological order
- stores the data so that one MINTIME may cross record boundaries

Figure 53 shows an example of how these records can be arranged on a Monitor III VSAM data set.

```
 Header
 and Index|  MINTIME 1 | MINTIME 2 | MINTIME n |
|*********|************|***********|....********|
|         |           |          |          |          |
|Record 1 |Record 2   |Record 3  |Record 4  |  .....   |Record n|
```

Figure 53. Monitor III Data Set Record

Record processing requires reading the header (record 1) and index to obtain the offset and length of a selected MINTIME set of samples. The record(s) containing the MINTIME sets of samples must be read into contiguous storage before RMF can process them. MINTIME 2 starts in record 3 and ends in record 4. Note that before MINTIME processing can begin, both records 3 and 4 must be read into contiguous storage.

End of Programming Interface information

Data set decompression

Programming Interface information

The MINTIME set-of-samples stored on VSAM data sets is compressed by RMF prior to storing the data. For direct access of the VSAM data sets and processing without use of the Monitor III reporter, you will need to use the Data Set Decompression Interface Service module, ERB3RDEC.

To use this service, the caller must invoke the module ERB3RDEC with the registers and parameter area described in “Parameter area contents” on page 199. The service returns only *one* record to the caller, which contains all the data.

Programming considerations

Do not link the module ERB3RDEC to your application program. Assembler programs must use LOAD or LINK macros to access the module; PL/I programs must use FETCH/RELEASE; and C programs must use the built-in function FETCH.

The caller must be in 31-bit addressing mode and can run unauthorized.

Registers at entry

The contents of the registers on entry to this service are:

Register	Contents
0	Reserved
1	Parameter list address
2-12	Reserved
13	Standard save area address
14	Return address

15 Entry point address of ERB3RDEC

Parameter area contents

The parameter area passed by the caller to the RMF Data Set Decompression Interface Service is a 3-fullword string, preceded by a halfword containing the length of the parameter area. The parameter area is as follows:

First word Bytes 0 to 3: address of the compressed set-of-samples

Second word Bytes 4 to 7: address of output area for decompressed set-of-samples

Third word Bytes 8 to 11: length of output area

Output

ERB3RDEC returns the following information in the parameter area depending on the return code (RC):

Third word

RC=0: length of the output area for the decompressed set-of-samples.

RC=4: minimum length required for the output area to hold the decompressed set-of samples.

RC>4: the bytes remain unchanged.

Return codes

Upon return from this service, register 15 provides return codes listed in Table 16.

Table 16. Return Codes for the Data Set Decompression Interface Service

Return Code (Decimal)	Description
0	Decompression successful, length of decompressed set-of-samples returned.
4	Decompression unsuccessful. The output area was too small to hold the decompressed set-of-samples. The minimum length required to hold the decompressed set-of-samples is returned. Obtain a larger output area and try again.
8	Decompression unsuccessful. Address passed for the compressed set-of-samples points to an uncompressed set-of-samples.
12	Decompression unsuccessful. Address passed for the compressed set-of-samples does not point to a valid set-of-samples.

Coded example

The following Assembler code example calls the Data Set Decompression Interface Service twice. The first call obtains the required length of the output area for the specified decompressed set-of samples. The second call performs the decompression.

This sample code assumes that register 2 points to the address of the compressed set-of-samples. It can be included in your installation's data retrieval code.

```
* Assuming, register 2 points to the compressed set-of-samples
         MVC      INRECA,0(R2)         Pointer to input record
* Calls Decompress Routine to retrieve the length of the
* uncompressed record.
         LA       R1,OUTAREA           Address of uncompressed record
```

```
         ST       R1,OUTRECA          Stores address in parmlist
         MVC      OUTRECL,INITLNG     Length of uncompressed record
         LA       R1,PARMADDR         Parameter to R1
         LINK     EP=ERB3RDEC         Invokes decompress routine
* Checks Return Code
         ST       R15,RETCODE         Saves return code
         CLC      R15,=F'4'           Checks return code
         BNE      PROCESS             Output area NOT too small
* Allocates required output area
         L        R3,OUTRECL          Required output length
         SR       R4,R4               Subpool 0
         GETMAIN  RU,LV=(3),SP=(4)    Get storage
         ST       R1,OUTRECA          Address of uncompressed record
* Calls Decompress Routine
         LA       R1,PARMADDR         Parameter to R1
         LINK     EP=ERB3RDEC         Invokes decompress routine
* Checks Return Code
         ST       R15,RETCODE         Saves return code
         LTR      R15,R15             Tests return code
         BZ       PROCESS             Decompress successful
* Decompress not successful. Releases output area
         L        R2,OUTRECA          Area address
         L        R3,OUTRECL          Area length
         SR       R4,R4               Subpool 0
         FREEMAIN RU,LV=(3),A=(2),,SP=(4)
PROCESS  DS       0H
* Check return code and process the decompressed record here.
* OUTRECA contains the address of the uncompressed record and the
* return code from ERB3RDEC is in RETCODE.
                  ...
* Declarations for the coding example above
INITLNG  DC       F'100'              Initial length
OUTAREA  DS       CL100               Initial output area
PARMADDR DC       A(PARMLIST)         Address of parameter list
RETCODE  DS       F                   Return code
         CNOP     2,4                 Alignment
PARMLIST DC       H'12'               Length of parameter area. This
*                                     field has to be initialized
*                                     with the decimal value 12.
INRECA   DS       F                   First word. It has to be
*                                     initialized with the address of
*                                     the compressed set-of-samples.
OUTRECA  DS       F                   Second word. It has to be
*                                     initialized with the address of
*                                     the output area which holds the
*                                     uncompressed set-of-samples.
OUTRECL  DS       F                   Third word. It has to be
*                                     initialized with the size of
*                                     the output area. ERB3RDEC will
*                                     return the size of the un-
*                                     compressed set-of-samples in
*                                     this field.

* Registers
R0       EQU      0
R1       EQU      1
R2       EQU      2
R3       EQU      3
R4       EQU      4
R5       EQU      5
R6       EQU      6
R7       EQU      7
R8       EQU      8
R9       EQU      9
R10      EQU      10
R11      EQU      11
```

```
R12      EQU      12
R13      EQU      13
R14      EQU      14
R15      EQU      15
```

End of Programming Interface information

Data set content

Programming Interface information

A MINTIME set of samples collected during the Monitor III gatherer session can be formatted and displayed during a Monitor III reporter display session. Each MINTIME set of samples is independent of other MINTIME sets of samples, and if you specify the same MINTIME value as that of the RANGE period for a display session, the report displays the information for that MINTIME set of samples collected during the gatherer session. Measurement values for each MINTIME set of samples are organized as tables or records, the formats of which appear at the end of this chapter. The types of measurement tables or records are:

ERBASIG3 ASID table
ERBCATG3 Cache data information table
ERBCFIG3 Coupling facility information table
ERBCPCDB CPC data control block
ERBCPDG3 Channel data information table
ERBCPUG3 Processor data control block
ERBCSRG3 Common storage remaining table
ERBDSIG3 Data set header and index
ERBDVTG3 Device table
ERBENCG3 Enclave data table
ERBENTG3 Enqueue name table
ERBGEIG3 General information table
ERBGGDG3 Global gatherer data table
ERBOPDG3 OMVS process data table
ERBRCDG3 Resource collection data table
ERBREDG3 Resource data record
ERBSHDG3 Sample header
ERBSSHG3 MINTIME set of samples header
ERBSPGG3 Storage group and volume data table
ERBSVPG3 Service policy data table
ERBUWDG3 USE/WAIT record
ERBXMHG3 Moved samples header control block

Each is described in "Monitor III data set record and table formats" on page 205. Each offset is from the beginning of the table that contains the offset. Clock times are local from the time-of-day (TOD) clock.

Figure 54 shows the relationships between the Monitor III data set support tables and records.

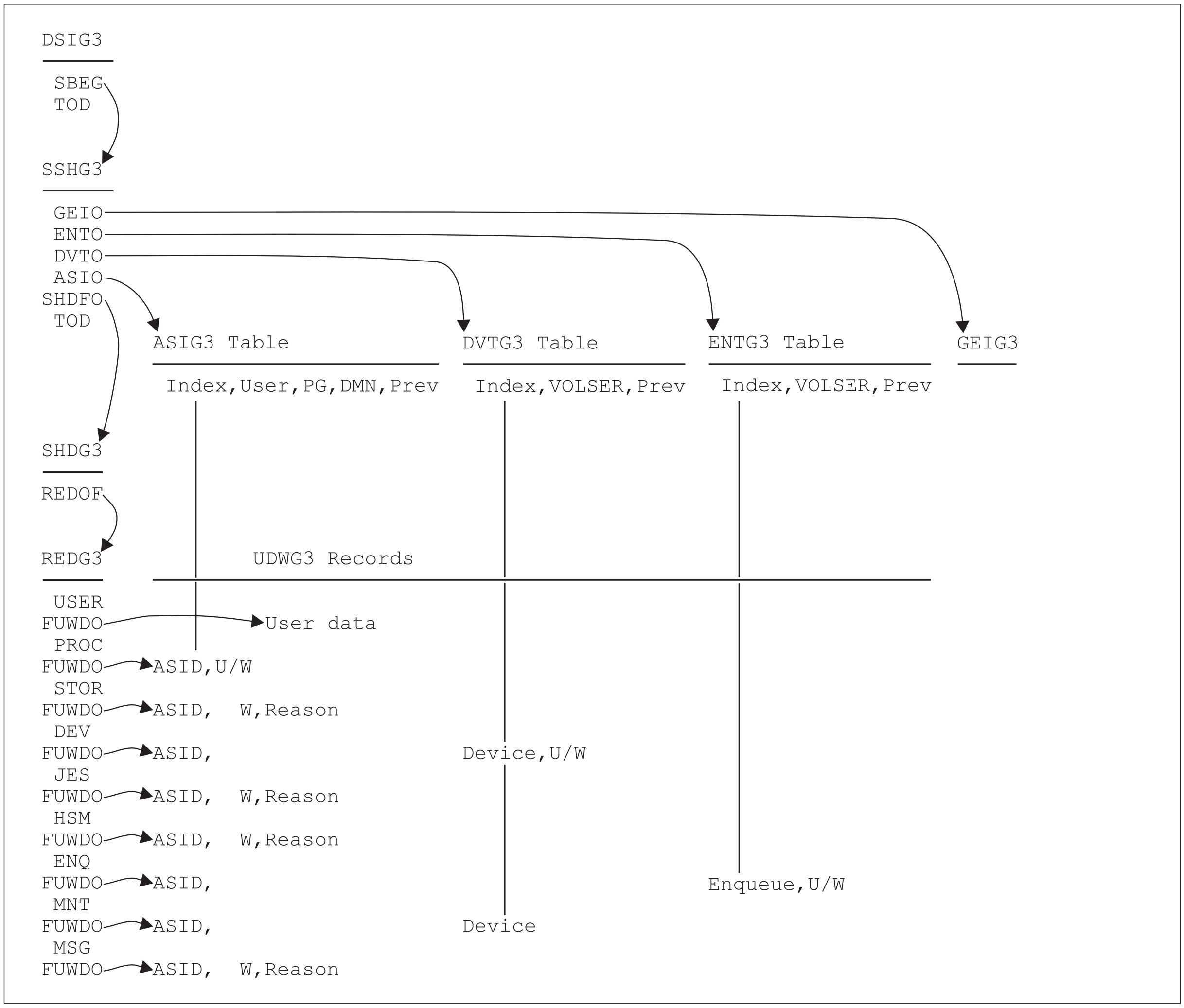


Figure 54. Monitor III Measurement Table and Record Relationships

The data set header and index (ERBDSIG3) describe the available measurement times (MINTIME sets of samples) and the data set offsets of each MINTIME set of samples header (ERBSSHG3).

The MINTIME set of samples header (ERBSSHG3) contains offsets to the address space id table (ERBASIG3), the device table (ERBDVTG3), enqueue name table (ERBENTG3), the general information table (ERBGEIG3), a group of sample headers (ERBSHDG3), and the common storage remaining table (ERBCSRG3). These tables describe information about each MINTIME interval within a data set.

Each sample header (ERBSHDG3) describes one sample CYCLE, and sample headers (ERBSHDG3) within one MINTIME are chained together by offsets.

The resource records (ERBREDG3) contain information about sampling for each resource. RMF first samples each type of hardware and software resource; RMF then samples user-written exit routines. The sample header (ERBSHDG3) for user-written exit routines contains an offset to the first resource record.

RMF creates in sequence one USE/WAIT record (ERBUWDG3) for each entry it finds in the queue for each resource. The resource record (ERBREDG3) contains an offset to the first USE/WAIT record for each resource.

The address space id table (ERBASIG3) contains one entry for each ASID/job combination. Each table entry contains the ASID number, its own index, and the index of the previous table entry for the ASID. (During one MINTIME interval, a job could exit, then reenter the system and therefore be assigned the same ASID. In this case, the job could have two sets of table entries for that MINTIME.)

The device table (ERBDVTG3) contains an entry for each device/VOLSER combination. Each entry contains the device number, its own index, and the index of the previous table entry for the device.

RMF correlates USE/WAIT records with their current table entries also by index.

To obtain the offset of each entry within the ASIG3 or DVTG3 table, multiply the length of each table entry by the index (see Figure 54).

```
Index x length of table entry.
```

For higher level languages, ASIG3 or DVTG3 arrays can be accessed with the index and an origin of 0.

To obtain the offset of each entry within the ENTG3 table, multiply the length of each table entry by the index (see Figure 54) minus 1:

```
(Index - 1) x length of table entry.
```

For higher level languages, the ENTG3 array can be accessed with the index and an origin of 1.

The common storage remaining table (ERBCSRG3) contains one entry for each job that ended and did not release all common storage. Each table entry contains the ASID number, the jobname, the JES-ID, the termination date, the termination time, and the amount of remaining common storage.

End of Programming Interface information

Monitor III data set record and table formats

Programming Interface information

This section describes the measurement records and tables used for the Monitor III data set support function. Fields that are reserved for RMF are used for debugging purposes, for maintaining the data areas, or do not contain RMF Monitor III report data.

Note: The following record and table mappings apply only to the current release and are subject to change for future releases.

ERBASIG3 - Address space identification table

Offsets		Name	Length	Format	Description
Dec	Hex				
ASIG3 Header Section:					
0	0	ASIASIG3	5	EBCDIC	Acronym 'ASIG3'
5	5	ASIVERG3	1	binary	Control block version X'12'
6	6	ASIHDRLE	1	binary	Length of ASIG3 header
7	7	*	1	*	Reserved
8	8	ASIENTMX	4	binary	Number of table entries
12	C	ASIENTNR	4	binary	Index of last table entry
16	10	ASIENTLN	4	binary	Length of one entry
20	14	ASISSTVO	4	binary	Offset to service-class-served table
24	18	*	8	*	Reserved
32	20	ASIENTRY(*)	*	*	Array of all ASID table entries
ASIG3 Table Entry Section:					
0	0	ASIENIDX	2	binary	Index of this table entry
2	2	ASIPREVI	2	binary	Index of the previous table entry for the same address space (ASID)
4	4	ASIJOBNA	8	EBCDIC	Jobname for this address space id (ASID). This and the next 5 offsets describe the sort criteria for the address space (ASID). RMF creates a new entry whenever the JOBNAME changes for the address space.
12	C	ASINPG	2	binary	Control performance group
14	E	*	1	*	Reserved
15	F	ASIDMNN	1	binary	Domain
16	10	ASIASINR	2	binary	ASID number

Offsets		Name	Length	Format	Description
Dec	Hex				
18	12	ASIFLAG1	2	binary	Job flags **Bit** **Meaning When Set** **0** Started task **1** Batch job **2** TSO ASID **3** ASCH ASID **4** OMVS ASID **5-15** Reserved
20	14	ASICPUTA	4	binary	Total TCB+SRB time (in milliseconds)[1]
24	18	ASIDCTIA	4	binary	Total channel connect time (in 128 microsecond units)[1]
28	1C	ASIFIXA_VE	4	floating point	Number of central fixed frames[1]
32	20	ASITRCA	4	binary	Total number of transactions[1]
36	24	ASIFMCT_VE	4	floating point	Number of frames for swapped-in users[1]
40	28	ASIFMCTI_VE	4	floating point	Number of frames for idle users[1]
44	2C	ASIESF_VE	4	floating point	Number of expanded storage frames for swapped-in users[1]
48	30	ASIESFI_VE	4	floating point	Number of expanded storage frames for idle users[1]
52	34	ASISMPCT	2	binary	Number of valid samples
54	36	ASISWAP	2	binary	Number of samples when job was physically swapped-out
56	38	ASIIDLE	2	binary	Number of samples when job was idle
58	3A	ASISWAR	2	binary	Number of samples when job was swapped-out ready
60	3C	ASIACT	2	binary	Active using or delayed count
62	3E	ASIUKN	2	binary	Number of samples when job status was unknown
64	40	ASISUSEN	2	binary	Number of single state using samples
66	42	ASISUCPR	2	binary	Number of single state samples using processor (PROC)
68	44	ASISUCDV	2	binary	Number of single state samples using device (DEV)
70	46	ASISWAIN	2	binary	Number of single state samples delayed by any resource
72	48	ASISDCPR	2	binary	Number of single state samples delayed by the processor (PROC)
74	4A	ASISDCDV	2	binary	Number of single state samples delayed by device (DEV)
76	4C	ASISDCST	2	binary	Number of single state samples delayed by paging or swapping (STOR)
78	4E	ASISDCJE	2	binary	Number of single state samples delayed by JES

Offsets		Name	Length	Format	Description
Dec	Hex				
80	50	ASISDCHS	2	binary	Number of single state samples delayed by HSM
82	52	ASISDCEN	2	binary	Number of single state samples delayed by ENQ
84	54	ASIVECTA	4	binary	Total accumulated vector processor time
88	58	ASISDCSU	2	binary	Number of single state samples delayed by SUBS
90	5A	ASISDCOP	2	binary	Number of single state samples delayed by OPER
92	5C	ASISDCMS	2	binary	Number of single state samples delayed by OPER MESSAGE
94	5E	ASISDCMT	2	binary	Number of single state samples delayed by OPER MOUNT
96	60	ASIPAGES	2	binary	Page delay
98	62	ASISWAPS	2	binary	Swap delay
100	64	ASIDIV_VE	4	floating point	Number of DIV frames
104	68	ASIAUXSC_VE	4	floating point	Number of auxiliary slots
108	6C	ASIPINA	4	binary	Page-in counts
112	70	ASIDIVCT	2	binary	Number of DIV invocations
114	72	ASIACTHF	2	binary	Number of address spaces active and holding storage counter
116	74	ASISWAPI	2	binary	Number of address spaces swapped in (not logically and not physically swapped
118	76	ASISDCXC	2	binary	Number of single state samples delayed by XCF - part of subs
120	78	ASIJCLAS	8	EBCDIC	Job class, Source: OUCBCLS
128	80	ASIPINES	4	binary	Expanded storage page-in count
132	84	ASIFLAG2	4	binary	Common storage flags **Bit** **Meaning When Set** **0** CSA amounts incomplete **1** SQA amounts incomplete **2** APPC initiator **3** BATCH initiator **4-31** Reserved
136	88	ASICSASC	4	binary	CSA sample count
140	8C	ASISQASC	4	binary	SQA sample count
144	90	ASICSAA	4	floating point	CSA allocation
148	94	ASISQAA	4	floating point	SQA allocation
152	98	ASIECSAA	4	floating point	ECSA allocation
156	9C	ASIESQAA	4	floating point	ESQA allocation
160	A0	ASIJLCYC	4	binary	Time-offset when this job was last found in the system, expressed in CYCLE time units.

Offsets		Name	Length	Format	Description
Dec	Hex				
164	A4	ASIJOBST	8	EBCDIC	Job selection time in GMT
172	AC	ASIJESID	8	EBCDIC	JES ID
180	B4	ASITET	4	binary	Transaction elapsed time, in 1024 microsecs units
184	B8	ASISRBTA	4	binary	Total accumulated SRB time
188	BC	ASIIOCNT	4	binary	IO count
192	C0	ASILSCT	2	binary	Count of "long" logical swaps
194	C2	ASIESCT	2	binary	Count of "long" swaps to expanded storage
196	C4	ASIPSCT	2	binary	Count of "long" physical swaps
198	C6	ASILSCF	4	floating point	Sum of all central frames for logically swapped user at all samples.
202	CA	ASILSEF	4	floating point	Sum of all expanded frames for logically swapped user at all samples.
206	CE	ASILSSA	2	binary	Total logically swapped samples
208	D0	ASIPSEF	4	floating point	Sum of all expanded frames for swapped user (except logical) at all samples.
212	D4	ASIPSSA	2	binary	Total swapped samples (except logical)
214	D6	ASIORTI	2	binary	STOR/OUTR delay samples for swap reason 1: Terminal input wait
216	D8	ASIORTO	2	binary	STOR/OUTR delay samples for swap reason 2: Terminal output wait
218	DA	ASIORLW	2	binary	STOR/OUTR delay samples for swap reason 3: Long wait
220	DC	ASIORXS	2	binary	STOR/OUTR delay samples for swap reason 4: Aux. storage shortage
222	DE	ASIORRS	2	binary	STOR/OUTR delay samples for swap reason 5: Real storage shortage
224	E0	ASIORDW	2	binary	STOR/OUTR delay samples for swap reason 6: Detected long wait
226	E2	ASIORRQ	2	binary	STOR/OUTR delay samples for swap reason 7: Requested swap
228	E4	ASIORNQ	2	binary	STOR/OUTR delay samples for swap reason 8: Enqueue exchange swap
230	E6	ASIOREX	2	binary	STOR/OUTR delay samples for swap reason 9: Exchange swap
232	E8	ASIORUS	2	binary	STOR/OUTR delay samples for swap reason 10: Uniliteral swap
234	EA	ASIORTS	2	binary	STOR/OUTR delay samples for swap reason 11: Transition swap
236	EC	ASIORIC	2	binary	STOR/OUTR delay samples for swap reason 12: Improve central storage usage

Offsets		Name	Length	Format	Description
Dec	Hex				
238	EE	ASIORIP	2	binary	STOR/OUTR delay samples for swap reason 13: Improve system paging rate
240	F0	ASIORMR	2	binary	STOR/OUTR delay samples for swap reason 14: Make room for an out too long user
242	F2	ASIORAW	2	binary	STOR/OUTR delay samples for swap reason 15: APPC wait
244	F4	ASIORIW	2	binary	STOR/OUTR delay samples for swap reason 16: OMVS input
246	F6	ASIOROW	2	binary	STOR/OUTR delay samples for swap reason 17: OMVS output
248	F8	ASIRCLX	2	binary	Report-class-list index
250	FA	ASIORSR	2	binary	STOR/OUTR delay samples for swap reason 18: In-real swap
252	FC	ASICPUC	2	binary	CPU capping delay
254	FE	ASIACOM	2	binary	Common paging
256	100	ASIAPRV	2	binary	Private paging
258	102	ASIAVIO	2	binary	VIO paging
260	104	ASIASWA	2	binary	Swapping
262	106	ASIUNKN	2	binary	Unknown count for calculating execution velocity
264	108	ASICCAP	2	binary	Resource capping delay
266	10A	ASICQUI	2	binary	Quiesce delay
268	10C	ASIAXM	2	binary	Cross-memory delay
270	10E	ASIAHSP	2	binary	Hiperspace™ delay
272	110	ASICUSE	4	binary	CPU using
276	114	ASITOTD	4	binary	Total delays for calculating execution velocity
280	118	ASISRVO	4	binary	Offset from service-class-served table-header to corresponding row
284	11C	ASITOTSV	4	floating point	Total number of shared page views in this address space
288	120	ASISVINR	4	floating point	Total number of shared pages in central storage that are valid for this address space
292	124	ASISPVLC	4	floating point	Total number of shared page validations in this address space
296	128	ASIGSPPI	4	floating point	Total number of shared page-ins from auxiliary storage for this address space
300	12C	ASIGASPD	2	binary	Number of single state samples delayed for shared storage paging
302	12E	*	2	*	Reserved
304	130	ASIOREPL	4	binary	Number of outstanding replies
308	134	ASITOTU	4	binary	Number of multi-state using samples

Offsets		Name	Length	Format	Description
Dec	Hex				
312	138	ASIIOU	4	binary	Number of multi-state I/O using samples
316	13C	ASIASSTA	4	binary	Additional SRB time
320	140	ASIPHTMA	4	binary	Preemptable-class SRB time
324	144	ASIMSTS	4	binary	Miscellaneous states. **Bit** **Meaning When Set** **0** Address space is OMVS related **1** Address space matched a classification rule in the active policy which prevents managing the region based on the response time goals of its served transactions **2** CPU protection was assigned either to the address space or to transaction service classes being served by the space, and SRM is honoring the protection **3** Storage protection was assigned either to the address space or to transaction service classes being served by the space, and SRM is honoring the protection **4** This address space provides service to transactions classified to a different class than the address space itself **5** WLM is managing this address space to meet the goals of work in other service classes **6** Address space is a CICS TOR that matched a classification rule in the active policy which allows managing the region based on the region goals but also ensures that completed transactions are reported and used for management of the CICS AORs. **7-31** Reserved
328	148	ASISUCIF	2	binary	Number of single state samples using zAAP
330	14A	ASISUCIC	2	binary	Number of single state samples using zAAP on CP
332	14C	ASISDCIF	2	binary	Number of single state samples delayed by zAAP
334	14E	ASISDCCP	2	binary	Number of single state samples delayed by standard CP

Offsets		Name	Length	Format	Description
Dec	Hex				
336	150	ASICPTA	4	binary	Accumulated CPU time
340	154	ASIIFATA	4	binary	Accumulated zAAP time
344	158	ASIIFCTA	4	binary	Accumulated zAAP on CP time
348	15C	ASIMCUSE	4	binary	Multi state processor using count
352	160	ASIMCDLY	4	binary	Multi state processor delay count
356	164	ASISUCCP	2	binary	Number of single state samples using standard CP
358	166	ASISUCSP	2	binary	Number of single state samples using zIIP
360	168	ASISUCSC	2	binary	Number of single state samples using zIIP on CP
362	16A	ASISDCSP	2	binary	Number of single state samples delayed by zIIP
364	16C	ASI_TIME_ON_ZIIP	4	binary	Accumulated zIIP time
368	170	ASI_ZIIP_TIME_ON_CP	4	binary	Accumulated zIIP on CP time
372	174	ASI_IFA_PHTM	4	binary	zAAP-only equivalent of ASIPHTMA
376	178	ASI_ZIIP_PHTM	4	binary	zIIP-only equivalent of ASIPHTMA
380	17C	ASI_LargeMemoryObjects	4	floating point	Number of large memory objects allocated
384	180	ASI_LargePagesBackedInReal	4	floating point	Number of 1 MB frames backed in real storage
388	184	ASI_LVNMOMB	4	floating point	Number of private storage memory objects allocated
392	188	ASI_HVCommonNMOMB	4	floating point	Number of 64-bit common memory objects allocated
396	18B	ASI_LVSHRNMOMB	4	floating point	Number of shared memory objects allocated
400	190	ASI_LVABytes	8	floating point	Amount of private storage allocated from large virtual private memory in memory objects
408	198	ASI_HVCommonBytes	8	floating point	Amount of 64-bit common storage allocated
416	1A0	ASI_LVSHRBytes	8	floating point	Amount of shared storage allocated from large virtual memory in memory objects
424	1A8	ASI_HVVCommonHWMBytes	8	floating point	High water mark for the amount of 64-bit common storage allocated
432	1B0	ASI_LVMemLim	8	floating point	Address space memory limit in MB
[1] Sum of all values obtained at each sample. To obtain average values, divide by the number of valid samples (ASISMPCT).					

ERBCATG3 - Cache data information table

Offsets		Name	Length	Format	Description
Dec	Hex				
CATG3 Header Section:					
0	0	CATG3_Acro	5	EBCDIC	Acronym 'CATG3'
5	5	CATG3_Ver	1	binary	Control block version X'01'
6	6	*	2	*	Reserved
8	8	CATG3_Tot_Len	4	binary	Total length including this header, the maximum number of CATG3 array entries and all SMF 74 subtype 5 records
12	C	*	4	*	Reserved
16	10	CATG3_Hdr_Len	2	binary	Length of CATG3 header
18	12	CATG3_Entry_Len	2	binary	Length of one CATG3 array entry
20	14	CATG3_Entry_Num	4	binary	Number of CATG3 array entries in use
24	18	CATG3_Max_Num	4	binary	Maximum number of CATG3 array entries
28	1C	*	20	*	Reserved
48	30	CATG3_LDTO	8	binary	Offset GMT to local time (STCK format)
56	38	*	8	*	Reserved
CAT Table (CATG3) Array Entry:					
0	0	CATG3_SSID	2	binary	Cache subsystem ID (SSID)
2	2	CATG3_MDL	1	binary	Subsystem model
3	3	*	1	*	Reserved
4	4	CATG3_B_TOD	8	binary	Cache interval GMT start time (STCK format)[1]
12	C	CATG3_E_TOD	8	binary	Cache interval GMT end time (STCK format)[1]
20	14	CATG3_Msg_Flg	2	binary	If bit 8 is set, no SMF 74.5 record data available for this SSID
22	16	CATG3_Stat_Code	2	binary	Status code. 0 = SMF 74.5 record data available 4 = IOS return code given 8 = IDCSS01 return code given 98 = System or User Abend given
24	18	CATG3_RC_IOS	2	binary	IOS return code
26	1A	CATG3_RC_IDCSS	2	binary	IDCSS01 return code
28	1C	*	1	*	Reserved
29	1D	CATG3_CMPC	3	binary	Bit 0-11: System completion code Bit 12-23: User completion code
32	20	CATG3_Rec_VF	4	binary	Offset to SMF 74 subtype 5 record with Cache Subsystem Activity data
36	24	CATG3_Rec_Lng	4	binary	Length of SMF 74 subtype 5 record

[1] Device reserve activity can cause a data gatherer interface to wait until a RESERVE has been released. This in turn can cause the cache interval to be longer than the set of samples interval (see SSHTIBEG and SSHTIEND). To convert this value to local time, add CATG3_LDTO (see Header Section).

ERBCFIG3 - Coupling facility information table

Offsets		Name	Length	Format	Description
Dec	Hex				
CFIG3 Header Section:					
0	0	CFICFIG3	5	EBCDIC	Acronym 'CFIG3'
5	5	CFIVERG3	1	binary	Control block version X'05'
6	6	CFIHDRLE	2	binary	Length of CFIG3 header
8	8	CFITOTLE	4	binary	Length of total CFIG3
12	C	CFIENTLE	2	binary	Length of one CFI entry
14	E	CFIENTNR	2	binary	Number of CFI entries
16	10	CFISTROF	4	binary	Offset to first structure entry CFISTRUS
20	14	CFISTRLE	2	binary	Length of one structure entry
22	16	CFISTRNR	2	binary	Number of structure entries
24	18	CFISTEOF	4	binary	Offset to first structure extension entry CFISTRES. If extensions exist, there is one CFISTRES entry for a CFISTRUS entry.
28	1C	CFISTELE	2	binary	Length of one structure entry
30	1E	CFISTENR	2	binary	Number of structure entries
32	20	CFICONOF	4	binary	Offset to first structure connection entry CFICONNS
36	24	CFICONLE	2	binary	Length of one CFICONNS entry
38	26	CFICONNR	2	binary	Number of CFICONNS entries
40	28	CFIRANGE	4	binary	Reporting range in seconds
44	2C	CFIPONAM	8	EBCDIC	Policy name
52	34	CFIPOACT	8	EBCDIC	Policy activation time
60	3C	CFIPREQS	4	binary	Policy is not large enough to contain all structures that exist in the CF
64	40	CFIPREQC	2	binary	Policy is not large enough to contain all connections
66	42	*	18	*	Reserved
CFI Table (CFIG3) Entry Section:					
0	0	CFIENNAM	8	EBCDIC	Name of CF
8	8	CFIENSYS	8	EBCDIC	Name of system (from SYSNAME parameter of IEASYSxx member)
16	10	CFIENST1	4	binary	Index of first CFISTRUS/CFISTRES entries of this CF
20	14	CFIENST#	4	binary	Number of CFISTRUS/CFISTRES
24	18	CFIENLVL	4	binary	CFLEVEL of microcode
28	1C	CFIENMOD	4	EBCDIC	CF model
32	20	CFIENVER	3	EBCDIC	CF version
35	23	CFIENPAM	1	binary	Path used mask for CFIENPID

Offsets		Name	Length	Format	Description
Dec	Hex				
36	24	CFIENFLG	2	binary	Status flags **Bit** **Meaning When Set** **0** Coupling Facility was connected to the system at the end of the MINTIME **1** Coupling Facility became active during the MINTIME **2** Coupling Facility structure hardware data gathered by RMF master gatherer on this member **3** Coupling Facility structure hardware data gathered with CFDETAIL option **4** Coupling Facility structure hardware data snapshot values taken from SMF records **5** The CF storage is volatile when this bit = 1 **6** Policy change pending which will delete this coupling facility from the CFRM active policy when all allocated structures are gone from this coupling facility **7** The coupling facility to CFRM policy reconcile process is in progress **8** The coupling facility has failed **9** CF dynamic dispatching **10** Recovery manager active **11** Maintenance mode active **12-15** Reserved
38	26	CFIENSCG	2	binary	Number of subchannels defined in the I/O gen
40	28	CFIENSCU	2	binary	Number of subchannels currently in use
42	2A	CFIENSCL	2	binary	Number of subchannels that can be used (limit)
44	2C	CFIENPID(8)	1	binary	CF links. Valid entries have corresponding bit in CFIENPAM set
52	34	CFIENPNR	4	binary	Number of online processors
56	38	CFIENBSY	4	binary	Busy time in milliseconds
60	3C	CFIENWAI	4	binary	Wait time in milliseconds
64	40	CFIENPBC	4	binary	Number of times CF requests failed due to path busy
68	44	CFIENSCC	4	binary	Subchannel contention count (all subchannel busy)
72	48	CFIENTSD	4	binary	Total amount of CF storage defined (units = 4K byte blocks)
76	4C	CFIENTSF	4	binary	Amount of free CF storage (units = 4K byte blocks)
80	50	CFIENTOR	4	binary	Number of total requests

Offsets		Name	Length	Format	Description
Dec	Hex				
84	54	CFIENTID(8)	1	binary	CF link type. Valid entries have corresponding bit in CFIENPAM set
92	5C	CFIENSCN	4	binary	Connected MVS system counts - Number of systems connected to specified CF
96	60	CFIENSTI	4	binary	Structure count in policy - Number of records for structures in specified CF
100	64	CFIENSTO	4	binary	Structure count out policy - Number of structures in this CF which cannot be added to the policy
104	68	CFIENTCS	4	binary	Total control space in 4K blocks. Control space + non-control space = total space
108	6C	CFIENFCS	4	binary	Total free control space in 4K blocks
112	70	CFIENTDS	4	binary	Total dump space in 4K blocks
116	74	CFIENFDS	4	binary	Free dump space in 4K blocks
120	78	CFIENSCT	8	binary	Summed contention time in microseconds for waiting for subchannels to become free for synchronous immediate operations
128	80	CFIENFOC	4	binary	Count of the number of summed times - for unsuccessful operations
132	84	CFIENFOT	8	binary	Summed service time in microseconds of unsuccessful operations
140	8C	CFIENPDE	2	binary	Number of dedicated processors
142	8E	CFIENPSH	2	binary	Number of shared processors
144	90	CFIENPWG	2	binary	Shared processor average weight
146	92	*	2	*	Reserved
CFISTRUS Table Entry Section:					
0	0	CFISTNAM	16	EBCDIC	Name of connected structure in this CF
16	10	CFISTVER	8	binary	Structure Version number
24	18	CFISTTYP	1	binary	Structure type: 1 = unserialized list 2 = serialized list 3 = lock 4 = cache 5 = unknown
25	19	*	3	*	Reserved
28	1C	CFISTFLG	1	binary	Status Flags **Bit** **Meaning When Set** **0** Structure was connected to the system at the end of the MINTIME **1** Structure became active during the MINTIME **2-7** Reserved
29	1D	*	1	*	Reserved
30	1E	CFISTEIN	2	binary	Index of according CFI table entry
32	20	CFISTTRC	4	binary	Total number of user requests completed

Offsets		Name	Length	Format	Description
Dec	Hex				
36	24	CFISTARC	4	binary	Count of number of times for async. requests executed by CF
40	28	CFISTATM	8	binary	Summed service time for asynchronous requests in microseconds
48	30	CFISTSRC	4	binary	Count of number of times for sync. requests executed by CF
52	34	CFISTSTM	8	binary	Summed service time for synchronous requests in microseconds
60	3C	CFISTSTA	4	binary	Number of requests changed from synchronous to asynchronous
64	40	CFISTQRC	4	binary	Count of number of times for queued requests
68	44	CFISTDRC	4	binary	Number of times a request was found delayed in case of dump serialization
72	48	CFISTCN	4	binary	Lock structure only: Number of times any request encountered lock contention
76	4C	CFISTFCN	4	binary	Lock structure only: Number of times any request encountered false lock contention (storage contention within the structure)
80	50	CFISTCOM	2	binary	Maximum number of connections allowed when structure was allocated in CF
82	52	*	2	*	Reserved
84	54	CFISTCOT	4	binary	Total of connections to the specified structure
88	58	CFISTCOP	4	binary	Number of connections to the specified structure with problems
92	5C	CFISTQTM	8	binary	Summed service time for queued requests in microseconds
100	64	CFISTFLE	2	binary	Status flags extended. **Bit** **Meaning When Set** **0** Only one structure allocated with this name **1** Rebuild (old): The original active structure is now the old structure **2** Rebuild (new): This structure is the new structure **3** Transitional state **4** Hold state **5** Structure cannot be deallocated since a dump table is associated with it **6** Structure failure for this version of the structure **7** Structure allocated with STRDISP = KEEP **8** Change pending in structure policy **9** Structure is defined in policy **10** Structure contains users **11-15** Reserved

Offsets		Name	Length	Format	Description
Dec	Hex				
102	66	CFISTRBP	1	binary	REBUILDPERCENT as specified in active CFRM policy
103	67	*	3	*	Reserved
106	6A	CFISTPL	35	EBCDIC	CF preference list
141	8D	CFISTXL	67	EBCDIC	Structure exclusion list
208	D0	CFISTETM	8	floating point	Summed structure execution time in microseconds
CFISTRES Table Entry Section:					
0	0	CFIENCN1	4	binary	Index of first CFICONNS entry belonging to this structure
4	4	CFIENCN#	4	binary	Number of CFICONNS entries belonging to this structure
8	8	CFISTSIZ	4	binary	Allocated size of structure (units = 4K byte blocks)
12	C	CFISTMAE	4	binary	List structure only: Maximum number of elements
16	10	CFISTCUE	4	binary	List structure only: Current number of elements in use
20	14	CFISTLEL	4	binary	List + Lock structure only: Limit on number of list entries
24	18	CFISTLEM	4	binary	List + Lock structure only: Maximum number of list entries used during MINTIME
28	1C	CFISTLTL	4	binary	Lock + serialized List structure only Limit on number of lock table entries
32	20	CFISTLTM	4	binary	Lock + serialized List structure only Maximum number of lock table entries used during MINTIME
36	24	CFISTDEN	4	binary	Cache structure only: Total directory entry count
40	28	CFISTDEL	4	binary	Cache structure only: Total data element count
44	2C	CFISCDEC	4	binary	Cache structure only: Current directory entry count
48	30	CFISCDAC	4	binary	Cache structure only: Current data element count
52	34	CFISCRHC	4	binary	Cache Read hit Counter
56	38	CFISCWHC	4	binary	Cache Write hit counter
60	3C	CFISCDER	4	binary	Cache Directory entry reclaim counter
64	40	CFISCXIS	4	binary	Cache XI counter
68	44	CFISCCOC	4	binary	Cache Castout Counter
72	48	CFISTXSS	4	binary	Maximum structure size (units = 4K byte blocks)
76	4C	CFISTMSS	4	binary	Minimum structure size (units = 4K byte blocks)
80	50	CFISTDTS	4	binary	Structure dump table size (4k blocks)

Offsets		Name	Length	Format	Description
Dec	Hex				
84	54	CFISTLHD	4	binary	Number of list headers (list only)
88	58	CFISTLEC	1	binary	List element characteristic. Size of list element in bytes is 256*(2**LELX)
89	59	CFISTMDS	1	binary	Maximum data list entry size (maximum number of elements per entry)
90	5A	CFISTLTC	1	binary	Lock table entry characteristic
91	5B	CFISTLFL	1	binary	List structure flags
92	5C	CFISTDEC	1	binary	Data area element characteristic (cache only)
93	5D	CFISTDAS	1	binary	Maximum data area size (cache only)
94	5E	CFISTDSC	1	binary	Maximum storage class (cache only)
95	5F	*	1	*	Reserved
96	60	CFISTDCC	2	binary	Maximum castout class (cache only)
98	62	CFISTFCC	2	binary	First castout class (cache only)
100	64	CFISTLCC	2	binary	Last castout class (cache only)
102	66	CFISTCFL	1	binary	Cache structure flags
CFICONNS Table Entry Section:					
0	0	CFICNSIN	2	binary	Index of according CFISTRUS table entry
2	2	*	2	*	Reserved
4	4	CFICNSYS	8	EBCDIC	Name of connecting system
12	C	CFICNNAM	16	EBCDIC	Name of connection
28	1C	CFICNJOB	8	EBCDIC	Name of connecting job
36	24	CFICNASI	2	binary	ASID of connecting job
38	26	CFICNSTA	2	binary	Connection status: 0 = Not known 1 = Active 2 = Failed Persistent 3 = Failing 4 = Disconnecting
40	28	CFICNCFL	4	binary	CF Level requested for connect
44	2C	CFICNFLG	2	binary	Connection flags **Bit** **Meaning When Set** **0** Rebuild allowed **1** Duplexing rebuild allowed **2** Structure alter allowed **3** Auto allowed **4** Auto allowed and suspend specified for connection. Valid only when ALLAuto=ON. Applicable only when CONN_Status = 1 . **5-15** Reserved.

ERBCPCDB - CPC data control block

Offsets		Name	Length	Format	Description
Dec	Hex				
CPCDB Header Section:					
0	0	CPC_EyeCt	5	EBCDIC	Name of CPCDB
5	5	CPC_VerNum	1	binary	Control block version X'05'
6	6	*	2	*	Reserved
8	8	CPC_HdrLen	4	binary	Length of CPCDB header
12	C	CPC_TotLen	4	binary	Total length of CPCDB
16	10	CPC_Flags	2	binary	Status flags **Bit** **Meaning When Set** **0** No data from Sysevent REQLPDAT **1** Partition PHYSICAL exists **2** Data invalid **3-15** Reserved
18	12	CPC_MaxLpars	2	binary	Maximum number of LPARs
20	14	CPC_MaxProcs	2	binary	Maximum number of processors
22	16	CPC_PhysProcs	2	binary	Number of physical processors
24	18	CPC_Homeo	4	binary	Offset to home LPAR Section
28	1C	CPC_Homel	2	binary	Length of home LPAR Section
30	1E	CPC_LparMainL	2	binary	Length of CPC LPAR Section
32	20	CPC_LparO	4	binary	Offset to CPC LPAR Sections
36	24	CPC_LparL	2	binary	Length of CPC LPAR Section with CPC Logical Processor Section(s)
38	26	CPC_LparN	2	binary	Number of CPC LPAR Sections
40	28	CPC_DTime	8	EBCDIC	Time delta between two DIAG calls
Home LPAR Section:					
0	0	CPC_HomeFlags	2	binary	Status flags **Bit** **Meaning When Set** **0** Capacity values available **1-2** Reserved **3** VARYCPU option set **4** WLM LPAR management enabled **5-14** Reserved **15** REQLPDAT version 3
2	2	*	2	*	Reserved
4	4	CPC_CecMSU	4	binary	Effective processor capacity available to the CPC
8	8	CPC_LparMSU	4	binary	see LPDatImgCapacity of IRALPDAT
12	C	*	4	*	Reserved
16	10	CPC_HomeLPName	8	EBCDIC	Name of the home partition
24	18	CPC_PhysAdj	4	binary	see LPDatPhyCpuAdjFactor of IRALPDAT
28	1C	CPC_WeightCumD	4	binary	see LPDatCumWeight of IRALPDAT. This is the delta between begin and end of MINTIME.

Offsets		Name	Length	Format	Description
Dec	Hex				
32	20	CPC_WeightNumD	4	binary	see LPDatWeightAccumCounter of IRALPDAT. This is the delta between begin and end of MINTIME.
36	24	*	2	*	Reserved
38	26	CPC_CapAdj	1	binary	Capacity adjustment indication
39	27	CPC_CapRsn	1	binary	Capacity change reason
40	28	CPC_ImgMsuLimit	4	binary	Image capacity MSU limit
44	2C	CPC_4hAverage	4	binary	see LPDatAvgImgService of IRALPDAT
48	30	CPC_UncappedTimeD	8	binary	Uncapped time delta. See LPDatCumUncappedElapsedTime of IRALPDAT. This is the delta between begin and end of MINTIME.
56	38	CPC_CappedTimeD	8	binary	Capped time delta. See LPDatCumCappedElapsedTime of IRALPDAT. This is the delta between begin and end of MINTIME.
64	40	CPC_MsuInterval	4	binary	Approximate time interval (in seconds) for each entry in the MSU table. see LPDatServiceTableEntryInterval of IRALPDAT.
68	44	CPC_MsuDataEntries	4	binary	Number of WLM intervals within the last 4 hours.
72	48	*	384	*	Reserved
456	1C8	CPC_GrpCapName	8	EBCDIC	Name of the capacity group to which the partition belongs
464	1D0	CPC_GrpCapLimit	4	binary	MSU limit for the capacity group to which this partition belongs
468	1D4	*	4	*	Reserved
472	1D8	CPC_GrpJoinedTOD	8	binary	Time when this LPAR has joined the group (STCK format)
CPC LPAR Section:					
0	0	CPC_LparName	8	EBCDIC	LPAR name
8	8	CPC_LparId	2	binary	LPAR number
10	A	CPC_LparFlags	1	binary	LPAR status flags **Bit** **Meaning When Set** **0** This is the home partition **1** LPAR data invalid **2** Number of processors defined for this partition exceeds limit **3** CPC_UPID is valid **4** Partition belongs to a capacity group; CPC_GroupName and CPC_GroupMLU are valid **5** WLM weight management enabled **6-7** Reserved
11	B	CPC_UPID	1	EBCDIC	User partition ID
12	C	CPC_LparDefMSU	4	binary	Defined MSU limit

Offsets		Name	Length	Format	Description
Dec	Hex				
16	10	CPC_OSname	8	EBCDIC	OS instance name
24	18	CPC_ProcO	4	binary	Offset to Logical Processor Sections
28	1C	CPC_ProcL	2	binary	Length of Logical Processor Section
30	1E	CPC_ProcN	2	binary	Number of Logical Processor Sections
32	20	CPC_LPCname	8	EBCDIC	LPAR cluster name
40	28	CPC_GroupName	8	EBCDIC	Name of the capacity group to which the partition belongs
48	30	CPC_GroupMLU	4	binary	Group maximum license units
52	34	CPC_OnlineCS	4	binary	Central storage (in MB) currently online to this partition
CPC Logical Processor Section:					
0	0	CPC_ProcId	2	binary	Logical CPU address
2	2	CPC_ProcTyp	1	binary	Processor type: 1=CP 2=ICF-pool 3=AAP 4=IFL 5=ICF 6=IIP
3	3	*	1	*	Reserved
4	4	CPC_ProcState	2	binary	Processor status indicators **Bit** **Meaning When Set** **0** Processor not available in MINTIME **1** Processor online **2** Processor dedicated **3** Wait completion = yes **4** Wait completion = no **5** Capping = yes **6** Polarization flag: this partition is vertically polarized; that is, HiperDispatch mode is active. CPC_ProcPolarWgt is valid. **7-8** Polarization indicator: **00** Horizontally polarized or polarization not indicated **01** Vertically polarized with low entitlement **10** Vertically polarized with medium entitlement **11** Vertically polarized with high entitlement **9-15** Reserved.

Offsets		Name	Length	Format	Description
Dec	Hex				
6	6	CPC_ProcChgInd	2	binary	Processor status-change indicators **Bit** **Meaning When Set** **0** Changed from online to offline or vice-versa **1** Changed from shared to dedicated or vice-versa **2** capping status changed **3** wait completion changed **4** Maximum weight changed **5-15** Reserved
8	8	CPC_ProcDispTimeD	8	binary	Dispatch time between begin and end of MINTIME in microseconds
16	10	CPC_ProcEffDispTimeD	8	binary	Effective dispatch time between begin and end of MINTIME in microseconds
24	18	CPC_ProcOnlineTimeD	8	binary	Online time between begin and end of MINTIME in microseconds
32	20	CPC_ProcMaxWeight	2	binary	Maximum LPAR share
34	22	CPC_ProcCurWeight	2	binary	Current LPAR share
36	24	CPC_ProcMinWeight	2	binary	Minimum LPAR share
38	26	CPC_ProcIniWeight	2	binary	Defined (initial) LPAR weight
40	28	CPC_ProcPolarWeight	4	binary	Weight for the logical CPU when HiperDispatch mode is active. See bit 6 of CPC_ProcState. Multiplied by a factor of 4096 for more granularity.
44	2C	*	4	*	Reserved

ERBCPDG3 - Channel data table

Offsets		Name	Length	Format	Description
Dec	Hex				
CPDG3 Header Section:					
0	0	CPDACR	5	EBCDIC	Acronym 'CPDG3'
5	5	CPDVER	1	EBCDIC	Control block version X'01'
6	6	CPDHDRL	2	binary	Length of CPDG3 header
8	8	CPDTOTL	4	binary	Total length of CPDG3
12	C	CPDNDAT	2	binary	Number of online channel path data sections
14	E	CPDLDAT	2	binary	Length of channel path data section
16	10	CPDODAT	4	binary	Offset to first channel path data section
Global channel information:					
20	14	CPDSMP	4	binary	Number of samples weighted by SRM, only valid if bit 6 of CPDFLG = OFF
24	18	CPDFLG	1	binary	Flags: **Bit** **Meaning When Set** **0** No data **1** Running in LPAR mode **2** CPMF available **3** Configuration change **4** DCM supported by hardware **5** Configuration contains DCM managed channels **6** HW allows multiple channel subsystems **7** Reserved
25	19	CPDCMI	1	binary	CPMF mode info: 0 = CPMF not available 1 = Compatibility mode 2 = Extended mode
26	1A	CPDCSS	1	binary	Channel subsystem ID, only valid if bit 6 of CPDFLG = ON
27	1B	CPDLPN	1	binary	Logical partition number
28	1C	CPDCFRC	4	binary	CPMF restart count
32	20	CPDCFSC	4	binary	CPMF sample count
36	24	*	12	EBCDIC	For alignment
Channel path performance data:					
0	0	CPDCPID	1	EBCDIC	Channel path identification
1	1	CPDCFLG	1	binary	Channel flags **Bit** **Meaning When Set** **0** Channel path is online **1** Channel path is shared between logical partitions **2** CPMF indication: this entry is invalid **3** Channel path is DCM managed **4** Channel characteristics changed **5-7** Reserved

ERBCPDG3 - Channel data table

Offsets		Name	Length	Format	Description
Dec	Hex				
2	2	CPDDFLG	1	binary	Channel data flags for channel measurement group 2 or 3 data **Bit** **Meaning When Set** **0** Channel characteristics word 1 valid **1** Channel characteristics word 2 valid **2** Channel characteristics word 3 valid **3** Channel characteristics word 4 valid **4** Channel characteristics word 5 valid **5-7** Reserved
3	3	CPDCPD	1	binary	Channel path description. For an explanation, you can issue the command: `D M=CHP`.
4	4	CPDCPA	5	EBCDIC	Channel path acronym
9	9	CPDCMG	1	binary	Channel measurement group
10	A	CPDCPP	1	binary	Channel path parameter
11	B	CPDGEN	1	binary	Channel type generation
12	C	CPDBSY	4	binary	Number of samples the channel path was busy, weighted by SRM
16	10	CPDPBY	4	binary	LPAR channel-path-busy time in units of 128 micro-seconds
20	14	CPDCPTX	4	binary	Extended last time stamp
20	14	*	1	binary	Overflow area
21	15	CPDCPTS	3	binary	Last CPMB entry time stamp in units of 128 micro-seconds (note that this time value wraps about every 35.79 minutes)
24	18	*	24	*	Reserved
48	30	CPDCCMC	20	EBCDIC	Characteristics part
68	44	CPDCCMD	28	EBCDIC	Measurements part
Channel measurement group 1 data - Characteristics part:					
0	0	CPDCC1	20	EBCDIC	not used
Channel measurement group 1 data - Measurements part:					
20	14	CPDTUT	4	floating point	CPMF total channel path busy time in units of 128 micro-seconds
24	18	CPDPUT	4	floating point	CPMF LPAR channel path busy time in units of 128 micro-seconds
28	1C	*	20	*	Reserved
Channel measurement group 2 data - Characteristics part:					
0	0	CPDMBC	4	floating point	CPMF maximum bus cycles per second (word 1)
4	4	CPDMCU	4	floating point	CPMF maximum channel work units per second (word 2)

Offsets		Name	Length	Format	Description
Dec	Hex				
8	8	CPDMWU	4	floating point	CPMF maximum write data units per second (word 3)
12	C	CPDMRU	4	floating point	CPMF maximum read data units per second (word 4)
16	10	CPDUS	4	floating point	CPMF data unit size in bytes (word 5)
Channel measurement group 2 data - Measurements part:					
20	14	CPDTBC	4	floating point	CPMF total bus cycle count
24	18	CPDTUC	4	floating point	CPMF total channel work unit count
28	1C	CPDPUC	4	floating point	CPMF LPAR channel work unit count
32	20	CPDTWU	4	floating point	CPMF total write data units
36	24	CPDPWU	4	floating point	CPMF LPAR write data units
40	28	CPDTRU	4	floating point	CPMF total read data units
44	2C	CPDPRU	4	floating point	CPMF LPAR read data units
Channel measurement group 3 data - Characteristics part:					
0	0	CPDPDU	4	floating point	CPMF LPAR data unit size in bytes (word 1)
4	4	CPDTDU	4	floating point	CPMF total data unit size in bytes (word 2)
8	8	CPDPUM	4	floating point	CPMF LPAR message sent unit size (word 3)
12	C	CPDTUM	4	floating point	CPMF total message sent unit size (word 4)
16	10	*	4	*	Reserved
Channel measurement group 3 data - Measurements part:					
20	14	CPDPMS	4	floating point	CPMF LPAR message sent units count
24	18	CPDTMS	4	floating point	CPMF total message sent units count
28	1C	CPDPUS	4	floating point	CPMF LPAR count of unsuccessful attempts to send messages
32	20	CPDPUB	4	floating point	CPMF LPAR count of unsuccessful attempts to receive messages due to unavailable buffers
36	24	CPDTUB	4	floating point	CPMF total count of unsuccessful attempts to receive messages due to unavailable buffers
40	28	CPDPDS	4	floating point	CPMF LPAR data units sent count
44	2C	CPDTDS	4	floating point	CPMF total data units sent count

ERBCPUG3 - Processor data control block

Offsets Dec	Offsets Hex	Name	Length	Format	Description
0	0	CPUG3_AC	5	EBCDIC	Name of CPUG3
5	5	CPUG3_VE	1	binary	Version of CPUG3
6	6	*	2	*	Reserved
8	8	CPUG3_HDRL	4	binary	Length of CPUG3 header
12	C	CPUG3_TOTL	4	binary	Total length this area
16	10	CPUG3_NUMPRC	8	binary	Number of processors (online during total MINTIME) multiplied by MINTIME (in microseconds)
24	18	CPUG3_LOGITI	8	binary	Logical CPU time in microseconds. This is the sum of MVS NON_WAIT time of all online logical processors in the time range
32	20	CPUG3_PHYSTI	8	binary	Physical CPU time in microseconds. This is the sum of all CPU times used by all logical processors. In the case of a native (non PR/SM) system this time is equal to the logical CPU time
40	28	CPUG3_STATUS	4	binary	Status information **Bit** **Meaning When Set** **0** BASIC mode system **1** LPAR mode system **2** Running as VM guest **3** Gatherer had permanent error **4** Diagnose 204 failed **5** VARY activity seen during the range. The number of logical processors used to accumulate the CPU time values varied. **6** Diagnose 204 extended format available **7** No MSU data available **8** Does not contain CPCDB data area **9-31** Reserved
44	2C	CPUG3_PRCON	4	binary	Number of online processors at end of MINTIME
48	30	CPUG3_NUMPRCOL	4	binary	Accumulated number of online processors. To get average number, divide by number of samples
52	34	CPUG3_NUMVECOL	4	binary	Accumulated number of online vector processors. To get average number, divide by number of samples
56	38	CPUG3_CPCOFF	4	binary	Offset to CPCDB area
60	3C	CPUG3_IFCON	4	binary	Number of zAAPs online at end of range
64	40	CPUG3_NUMIFCOL	4	binary	Accumulated number of zAAPs online. To get average number, divide by number of samples.
68	44	CPUG3_NUMPRIFA	8	binary	Accumulated online time of zAAPs in microseconds

Offsets		Name	Length	Format	Description
Dec	Hex				
76	4C	CPUG3_LOGITIFA	8	binary	Logical CPU time: The sum of MVS NON_WAIT time of all online logical zAAPs in the time range (in microseconds)
84	54	CPUG3_PHYSTIFA	8	binary	Physical CPU time: The sum of all CPU times used by all online logical zAAPs in the time range (in microseconds)
92	5C	CPUG3_SUCON	4	binary	Number of zIIPs online at end of range
96	60	CPUG3_NUMSUCOL	4	binary	Accumulated number of zIIPs online. To get average number, divide by number of samples.
100	64	CPUG3_NUMPRSUP	8	binary	Accumulated online time of zIIPs in microseconds
108	6C	CPUG3_LOGITSUP	8	binary	Logical CPU time: The sum of MVS NON_WAIT time of all online logical zIIPs in the time range (in microseconds)
116	74	CPUG3_PHYSTSUP	8	binary	Physical CPU time: The sum of all CPU times used by all online logical zIIPs in the time range (in microseconds)
124	7C	*	4	*	Reserved

ERBCSRG3 - Common storage remaining table

Offsets		Name	Length	Format	Description
Dec	Hex				
CSRG3 Header Section:					
0	0	CSRCSRG3	5	EBCDIC	Acronym ‘CSRG3’
5	5	CSRVERG3	1	binary	Control block version X'01'
6	6	*	1	*	Reserved
8	8	CSRHDRLE	2	binary	Length of CSRG3 header
10	A	CSRENTLE	2	binary	Length of one entry
12	C	*	4	*	Reserved
16	10	CSRENTNR	4	binary	Index of last available entry
20	14	*	12	*	Reserved
CSRG3 Table Entry Section:					
0	0	CSRASINR	2	binary	ASID number
2	2	*	2	*	Reserved
4	4	CSRJOBNA	8	EBCDIC	Jobname
12	C	CSRJESID	8	EBCDIC	JES-ID, taken from JSAB
20	14	CSRTDATE	4	EBCDIC	Ending Date, packed decimal OYYYYDDD, see documentation of the 'TIME' macro
24	18	CSRTTIME	4	EBCDIC	Ending Date, packed decimal HHMMSSth, see documentation of the 'TIME' macro
28	1C	CSRCSA	4	binary	CSA amount
32	20	CSRSQA	4	binary	SQA amount
36	24	CSRECSA	4	binary	ECSA amount
40	28	CSRESQA	4	binary	ESQA amount
44	2C	CSRFLAG	2	binary	Common Storage Flags **Bit** **Meaning** **0** CSA amounts complete **1** SQA amounts complete **2-15** Reserved
46	2E	*	2	*	Reserved

ERBDSIG3 - Data set header and index

Offsets		Name	Length	Format	Description
Dec	Hex				
DSIG3 Header Section:					
0	0	DSIDSIG3	5	EBCDIC	Acronym ‘DSIG3’
5	5	DSIGRMFV	1	binary	Control block version X'02'
6	6	DSIGID	4	EBCDIC	System identifier
10	A	*	2	*	Reserved
12	C	DSIGTODC	8	binary	Time data set was created
20	14	DSIGTODF	8	binary	Time stamp for first set of samples
28	1C	DSIGTODL	8	binary	Time stamp for last set of samples
36	24	DSIGFSPT	4	binary	Offset of first set of samples from ERBDSIG3
40	28	DSIGLSPT	4	binary	Offset of last set of samples from ERBDSIG3
44	2C	DSIGNEPT	4	binary	Offset of next set of samples to be written
48	30	DSIGFIPT	4	binary	Offset of the first index entry from ERBDSIG3
52	34	DSIGLIPT	4	binary	Offset of the last index entry from ERBDSIG3
56	38	DSIGNIPT	4	binary	Offset of next index to be written
60	3C	DSIGILEN	4	binary	Length of an index entry
64	40	DSIGINUS	4	signed	Number of current index to set of samples
68	44	DSIGTDSF	8	EBCDIC	Time stamp of first policy
76	4C	DSIGTDSL	8	EBCDIC	Time stamp of last policy
84	54	DSIGFPPT	4	signed	Offset to start of first policy
88	58	DSIGLPPT	4	signed	Offset to start of the last policy
92	5c	DSIGFPIP	4	signed	Offset to first policy index
96	60	DSIGLPIP	4	signed	Offset to last policy index
100	64	DSIGNPIP	4	signed	Offset to next policy index
104	68	DSIGCIPN	4	signed	Current index number to policy
108	6C	DSIGFIPN	4	signed	First index number to policy
112	70	DSIGSPLX	8	EBCDIC	Sysplex-ID of this system
120	78	DSIGSPXD	32	EBCDIC	Reserved for sysplex
152	98	*	104	*	Reserved
Data Set Index Section					
0	0	DSIGTOD1	8	EBCDIC	Time stamp for start of set of samples or service policy
8	8	DSIGTOD2	8	EBCDIC	Time stamp for end of set of samples or service policy
16	10	DSIGSBEG	4	binary	Offset from the start of the data set to the start of the set of samples or start of the service policy

Offsets		Name	Length	Format	Description
Dec	Hex				
20	14	DSIGSLEN	4	binary	Physical (possibly compressed) length of the set of samples or length of service policy as contained in SVPDLE
24	18	DSIGFLG	1	binary	Data set flags **Bit** **Meaning** **0** Service policy index **1-7** Reserved
25	19	*	3	*	Reserved

ERBDVTG3 - Device table

Offsets		Name	Length	Format	Description
Dec	Hex				
DVTG3 Header Section:					
0	0	DVTDVTG3	5	EBCDIC	Acronym 'DVTG3'
5	5	DVTVERG3	1	binary	Control block version X'08'
6	6	DVTHDRLE	1	binary	Length of DVTG3 header
7	7	DVTENTLE	1	binary	Length of each table entry
8	8	DVTENTMX	4	binary	Number of table entries
12	C	DVTENTNR	4	binary	Index of last table entry
16	10	DVTENTRY	104	EBCDIC	Entry in the device table
Device Table (DVTG3) Entry Section:					
0	0	DVTVOLI	6	EBCDIC	VOLSER for this device
6	6	DVTENIDX	2	binary	Index of this table entry
8	8	DVTDEVNR	2	binary	Device number in hexadecimal format
10	A	DVTPREVI	2	binary	Index of the previous table entry for the same device
12	C	DVTSMPCT	4	binary	Number of valid samples
16	10	DVTSMPNR	4	binary	Sample sequence number
20	14	DVTFLAG1	1	binary	Device type indicator **Bit** **Meaning When Set** **0** Reserved **1** DASD device **2** TAPE device **3** Number of alias exposures for a PAV device has changed **4** Virtual DASD **5** Reserved **6** LCU number is valid **7** Multiple exposure device (PAV)
21	15	DVTFLAG2	1	binary	Device storage indicators — these flags indicate if the time values in offsets 24 through 60 are available. **Bit** **Meaning When Set** **0** CONN/DISC/PEND time values at begin time available **1** CONN/DISC/PEND time values at end time available **2** DEV BUSY DELAY/CUB DELAY/DPB DELAY time values at end time available **3** DEV BUSY DELAY/CUB DELAY/DPB DELAY time values at end time available **4** Device has PLPA page data sets **5** Device has COMMON page data sets **6** Device has LOCAL page data sets **7** Reserved
22	16	DVTMEXNR	2	binary	Number of base and alias volumes

Offsets		Name	Length	Format	Description
Dec	Hex				
24	18	DVTDISIF	4	binary	Native device DISC time at the beginning of the MINTIME for this set of samples (in 2048-microsecond units)
28	1C	DVTPETIF	4	binary	Native device PEND time at the beginning of the MINTIME for this set of samples (in 2048-microsecond units)
32	20	DVTCOTIF	4	binary	Native device CONN time at the beginning of the MINTIME for this set of samples (in 2048-microsecond units)
36	24	DVTDVBIF	4	binary	Device busy delay time at the beginning of the MINTIME for this set of samples (in 2048-microsecond units)
40	28	DVTCUBIF	4	*	No longer used
44	2C	DVTDISIL	4	binary	Native device DISC time at the end of the MINTIME for this set of samples (in 2048-microsecond units)
48	30	DVTPETIL	4	binary	Native device PEND time at the end of the MINTIME for this set of samples (in 2048-microsecond units)
52	34	DVTCOTIL	4	binary	Native device CONN time at the end of the MINTIME for this set of samples (in 2048-microsecond units)
56	38	DVTDVBIL	4	binary	Device busy delay time at the end of the MINTIME for this set of samples (in 2048-microsecond units)
60	3C	DVTCUBIL	4	*	No longer used
64	40	DVTTYP	4	EBCDIC	Device type mapped by the UCBTYP macro
68	44	DVTIDEN	8	EBCDIC	Device identification (device model)
76	4C	DVTCUID	8	EBCDIC	Control unit model
84	54	DVTSPBIF	4	*	No longer used
88	58	DVTSPBIL	4	*	No longer used
92	5C	DVTIOQLC	4	binary	I/O queue length count
96	60	DVTSAMPA	4	binary	Accumulated I/O instruction count
100	64	*	2	*	Reserved
102	66	DVTLCUNR	2	binary	LCU number
104	68	DVTSAMPP	4	binary	I/O instruction count (previous value)
108	6C	DVTCMRIF	4	binary	Initial command response time first
112	70	DVTCMRIL	4	binary	Initial command response time last
116	74	DVTCUQTP	4	binary	Control unit queuing time previous sample
120	78	DVTCUQTN	4	binary	Accumulated control unit queuing time for devices not connected to FICON channel
124	7C	DVTCUQTF	4	binary	Accumulated control unit queuing time for devices connected to FICON channel
128	80	DVTHPNUM	4	signed	Accumulated number of HyperPAV aliases in each cycle
132	84	DVTPSM	4	signed	Number of successful PAV samples

Offsets		Name	Length	Format	Description
Dec	Hex				
136	88	DVTFLAG3	1	binary	Flag byte **Bit** **Meaning When Set** **0** device is a HyperPAV base device **1-7** Reserved
137	89	DVTHPCON	1	binary	Configured HyperPAV aliases for that LSS
138	8A	*	2	*	Reserved

ERBENCG3 - Enclave data table

Offsets		Name	Length	Format	Description
Dec	Hex				
ENCARRAY					
0	0	ENCG3ACR	5	EBCDIC	Acronym 'ENCG3'
5	5	ENCG3VER	1	binary	Control block version X'07'
6	6	*	2	*	Reserved
8	8	ENCG3TLN	4	binary	ENCG3 table length
12	C	ENCG3TET (6)	12	binary	table entry triplets
12	C	ENCG3TEO	4	binary	table entry offset
16	10	ENCG3TEL	4	binary	table entry length
20	14	ENCG3TEN	4	binary	table entry number
84	54	ENCG3DEO	4	binary	descriptor entry offset
88	58	ENCG3DEL	4	binary	descriptor entry length
92	5C	ENCG3DEN	4	binary	descriptor entry number
ENCG3 Header Section:					
0	0	ENCG3LEN	4	binary	table entry length
4	4	ENCTOKEN	8	EBCDIC	enclave token
12	C	ENCCLX	2	binary	service class index
12	C	ENCPGN	2	binary	performance group
14	E	ENCSRPG	2	binary	subsystem RCLX/RPGN
16	10	ENCNRPG	2	binary	trx name RPGN
18	12	ENCURPG	2	binary	userid RPGN
20	14	ENCCRPG	2	binary	trx class RPGN
22	16	ENCARPG	2	binary	account no RPGN
24	18	ENCPER	1	binary	SC|PG period
25	19	ENCDMN	1	binary	domain
26	1A	ENCG3KFI	1	binary	key field status flags **Bit** **Meaning When Set** **0** key SC/PG has changed **1** key period has changed **2** domain has changed **3-7** Reserved
27	1B	*	9	*	Reserved
36	24	ENCG3EDO	4	binary	offset from ENCG3 element to EDEG3 element
40	28	ENCG3SMP	4	binary	sample count
44	2C	ENCUSTOT	4	binary	using count Total
48	30	ENCDETOT	4	binary	delay count Total
52	34	ENCIDLES	4	binary	IDLE sample counts
56	38	ENCUNKNS	4	binary	UNKNOWN sample counts
60	3C	ENCUSCPU	4	binary	using count CPU
64	40	ENCDECPU	4	binary	delay count CPU

Offsets		Name	Length	Format	Description
Dec	Hex				
68	44	ENCDECCA	4	binary	delay count CPU capping
72	48	ENCDESTG	4	binary	delay count STOR paging
76	4C	ENCDECOM	4	binary	delay count COM paging
80	50	ENCDEXMM	4	binary	delay count X/M
84	54	ENCDESHP	4	binary	delay count Shared pag
88	58	ENCFLAGS	2	binary	ENCG3 descriptive flags **Bit** **Meaning When Set** **0** dependent enclave **1** original independent enclave **2** foreign independent enclave **3** foreign dependent enclave **4-15** Reserved
90	5A	ENCOASID	2	binary	Owner ASID
92	5C	ENCTOTS	4	binary	multistate samples
96	60	ENCUMCPU	4	binary	using count CPU (multistate samples)
100	64	ENCUMIO	4	binary	using count I/O
104	68	ENCDMCPU	4	binary	delay count CPU (multistate samples)
108	6C	ENCDMIO	4	binary	delay count I/O
112	70	ENCDMQUE	4	binary	delay count queue
116	74	ENCDMCCA	4	binary	delay count capping
120	78	ENCDMSTO	4	binary	delay count storage
124	7C	ENCMIDLE	4	binary	idle count
128	80	ENCMUNKN	4	binary	unknown count
132	84	ENCTCPUT	4	floating point	CPU time since creation of enclave
136	88	ENCCPUT	4	floating point	CPU time
140	8C	*	8	*	Reserved
148	94	ENCOWSYS	8	EBCDIC	Enclave owner system or blank if not a foreign enclave
156	9C	ENCOWJOB	8	EBCDIC	Enclave owner jobname or blank if not a foreign enclave
164	9C	ENCXTOK	32	EBCDIC	Enclave export token or zero if not a multi-system enclave
196	C4	ENCTIFAT	4	floating point	zAAP time since creation of enclave
200	C8	ENCTIFCT	4	floating point	zAAP on CP time since creation of enclave
204	CC	ENCIFAT	4	floating point	zAAP time
208	D0	ENCIFCT	4	floating point	zAAP on CP time
228	E4	ENCUSIFA	4	binary	using count zAAP
232	E8	ENCUSIFC	4	binary	using count zAAP on CP
236	EC	ENCDEIFA	4	binary	delay count zAAP
240	F0	ENCUMIFA	4	binary	using count zAAP (multistate samples)
244	F4	ENCUMIFC	4	binary	using count zAAP on CP (multistate samples)

Offsets		Name	Length	Format	Description
Dec	Hex				
248	F8	ENCDMIFA	4	binary	delay count zAAP (multistate samples)
252	FC	ENCUSCP	4	binary	using count CP (single state samples)
256	100	ENCDECP	4	binary	delay count CP (single state samples)
260	104	ENCTSUPT	4	floating point	zIIP time since creation of enclave
264	108	ENCTSUCT	4	floating point	zIIP on CP time since creation of enclave
268	10C	ENCSUPT	4	floating point	zIIP time
272	110	ENCSUCT	4	floating point	zIIP on CP time
276	114	*	16	*	Reserved
292	124	ENCUSSUP	4	binary	using count zIIP (single state sample)
296	128	ENCUSSUC	4	binary	using count zIIP on CP (single state sample)
300	12C	ENCDESUP	4	binary	delay count zIIP (single state sample)
304	130	ENCUMSUP	4	binary	using count zIIP (multi state sample)
308	134	ENCUMSUC	4	binary	using count zIIP on CP (multi state sample)
312	138	ENCDMSUP	4	binary	delay count zIIP (multi state sample)
RMF Enclave Descriptor Entry (EDEG3)					
0	0	EDETRXN	8	EBCDIC	transaction program name
8	8	EDEUSER	8	EBCDIC	userid
16	10	EDETRXC	8	EBCDIC	transaction class
24	18	EDENET	8	EBCDIC	netword id
32	20	EDELU	8	EBCDIC	logical unit name
40	28	EDEPLAN	8	EBCDIC	plan
48	30	EDEPCKG	8	EBCDIC	package
56	38	EDECNCTN	8	EBCDIC	connection
64	40	EDECOLL	18	EBCDIC	collection
82	52	EDECORR	12	EBCDIC	correlation
94	5E	ECDSUBT	4	EBCDIC	subsystem type
98	62	ECDFCN	8	EBCDIC	function name
106	6A	ECDSUBN	8	EBCDIC	subsystem name
114	72	EDESSPM	255	EBCDIC	subsystem parameter
369	171	EDEACCT	143	EBCDIC	accounting info
512	200	EDE_ PROCEDURENAME	18	EBCDIC	procedure name
530	212	EDE_PERFORM	8	EBCDIC	Perform=value
538	21A	*	2	*	Reserved
540	21C	EDE_PRIORITY	4	binary	Subsystem priority in binary format. Contains X'80000000' if the subsystem did not provide a priority.
544	220	EDE_ PROCESSNAME	32	EBCDIC	Process name
576	240	*	6	*	Reserved

ERBENTG3 - Enqueue name table

Offsets		Name	Length	Format	Description
Dec	Hex				
ENTG3 Header Section:					
0	0	ENTENTG3	5	EBCDIC	Acronym 'ENTG3'
5	5	ENTVERG3	1	binary	Control block version X'02'
6	6	ENTHDRLE	1	binary	Length of ENTG3 header
7	7	ENTENTLE	1	binary	Length of one entry
8	8	ENTENTMX	4	binary	Number of table entries
12	C	ENTENTNR	4	binary	Index of last filled entry (Highest possible index is ENTENTMX)
16	10	ENTENTRY (*)	48	EBCDIC	Entries in the ENTG3 table
ERBENTG3 Entry Section					
0	0	ENTENIDX	2	binary	ENQ NAME table entry index
2	2	ENTMAJNA	8	EBCDIC	Major name of this resource
10	A	ENTMINNA	36	EBCDIC	Minor name of this resource
46	2E	ENTSCOPE	1	binary	Scope of this resource **Bit** **Meaning When Set** **0** SYSTEM (When not set: NOSYSTEM) **1** SYSTEMS (When not set: NOSYSTEMS) **2** Reserved **3** GLOBAL (When not set: LOCAL) **4-7** Reserved
47	2F	ENTFLAGS	1	binary	Additional flags **Bit** **Meaning When Set** **0** This resource has suspended jobs. This flag is valid only during data gathering. It is not meaningful within reporter. **1-7** Reserved

ERBGEIG3 - General information table

Offsets		Name	Length	Format	Description
Dec	Hex				
0	0	GEIGEIG3	5	EBCDIC	Acronym 'GEIG3'
5	5	GEIVERG3	1	binary	Control block version X'10'
6	6	GEILEN	2	binary	Length of this control block (GEIG3)
8	8	*	16	*	Reserved
24	18	GEIVERSN	1	binary	CPU version number
25	19	*	1	*	Reserved
26	1A	GEIFLAG	1	binary	Processor flags **Bit** **Meaning When Set** **0** Service processor architecture supported **1** PR/SM machine **2** Reserved **3** BEG **4** END **5** No collector data **6** Data in GEIGG3 is unpredictable because ERB3GGSS terminated **7** No ENQ contention data available due to GRS system problem
27	1B	GEIFLG1	1	binary	Additional flags **Bit** **Meaning When Set** **0** No ENQ contention data available because of RMF/GRS interface problem **1** z/Architecture mode **2** CMR data available **3** zAAPs available **4** zIIPs available **5** Enhanced DAT architecture available **6-7** Reserved
28	1C	GEIMODEL	2	packed	CPU model number (The value is not signed.)
30	1E	GEIIPSID	2	EBCDIC	Installation performance specification (IPS) member suffix
32	20	GEIOPTN	2	EBCDIC	Option (OPT) member suffix
34	22	GEIICSN	2	EBCDIC	Installation control specification (ICS) member suffix
36	24	GEISID	4	EBCDIC	SYSTEM name (SMF system id)
40	28	*	4	*	Reserved
44	2C	GEIAHUIC_VE	4	floating point	Current system UIC[1]
48	30	GEIRPOOL_VE	4	floating point	Number of online real storage frames[1]
52	34	GEIRCOMA_VE	4	floating point	Number of real storage COMMON frames[1]
56	38	GEIRSQAA_VE	4	floating point	Number of real storage SQA frames[1]
60	3C	GEIRAFC_VE	4	floating point	Number of available real storage frames[1]

Offsets		Name	Length	Format	Description
Dec	Hex				
64	40	GEINUCA_VE	4	floating point	Number of nucleus (NUC) frames (real nucleus plus extended storage nucleus frames)[1]
68	44	GEIRSHR_VE	4	floating point	Number of real storage shared frames[1]
72	48	*	4	*	Reserved
76	4C	GEIEESPL_VE	4	floating point	Number of online extended storage frames[1]
80	50	GEIGAGE_VE	4	floating point	Extended storage migration age[1]
84	54	GEIECOME_VE	4	floating point	Number of extended storage COMMON frames[1]
88	58	GEIEAEC_VE	4	floating point	Number of available extended storage frames[1]
92	5C	*	4	*	Reserved
96	60	GEIESQAF_VE	4	floating point	Number of expanded storage SQA frames[1]
100	64	GEIRLPAF_VE	4	floating point	Number of central storage LPA frames[1]
104	68	GEIELPAF_VE	4	floating point	Number of expanded storage LPA frames[1]
108	6C	GEIRCSAF_VE	4	floating point	Number of central storage CSA frames[1]
112	70	GEIECSAF_VE	4	floating point	Number of expanded storage CSA frames[1]
116	74	GEIASMPC	4	binary	Monitor I sample count accumulated per MINTIME used by Monitor III reporter
120	78	GEIASQAO_VE	4	floating point	Number of SQA overflow frames - BEGIN of MINTIME used by Monitor III reporter[1]
124	7C	GEICSARE	4	binary	Amount of unallocated common area left (CSA + SQA)
128	80	*	4	*	Reserved
132	84	GEICPUON	2	binary	Snapshot number of online processors at end of the MINTIME[1]
134	86	*	2	*	Reserved
136	88	GEICSASZ	4	binary	IPL Size of CSA below 16M
140	8C	GEISQASZ	4	binary	IPL Size of SQA below 16M
144	90	GEIECSAZ	4	binary	IPL Size of CSA above 16M
148	94	GEIESQAZ	4	binary	IPL Size of SQA above 16M
152	98	GEISTCSA	4	binary	Start of CSA/ECSA tracking (first fullword of TOD)
156	9C	GEISTSQA	4	binary	Start of SQA/ESQA tracking (first fullword of TOD)
160	A0	GEIENCSA	4	binary	End of CSA/ECSA tracking (first fullword of TOD)
164	A4	GEIENSQA	4	binary	End of SQA/ESQA tracking (first fullword of TOD)
168	A8	GEINSCSA	4	binary	Number of CSA samples
172	AC	GEINSSQA	4	binary	Number of SQA samples
176	B0	GEICSAMX	4	binary	Max. allocated CSA below 16M
180	B4	GEISQAMX	4	binary	Max. allocated SQA below 16M

Offsets		Name	Length	Format	Description
Dec	Hex				
184	B8	GEIECSAX	4	binary	Max. allocated CSA above 16M
188	BC	GEIESQAX	4	binary	Max. allocated SQA above 16M
192	C0	GEICSASP	4	binary	Current allocated CSA below 16M
196	C4	GEISQASP	4	binary	Current allocated SQA below 16M
200	C8	GEIECSAP	4	binary	Current allocated CSA above 16M
204	CC	GEIESQAP	4	binary	Current allocated SQA above 16M
208	D0	GEICSAAV	4	floating point	Accumulated allocated CSA below 16M[1]
212	D4	GEISQAAV	4	floating point	Accumulated allocated SQA below 16M[1]
216	D8	GEIECSAV	4	floating point	Accumulated allocated CSA above 16M[1]
220	DC	GEIESQAV	4	floating point	Accumulated allocated SQA above 16M[1]
224	E0	GEICSACN	4	floating point	Accumulated CSA conv. below 16M[1]
228	E4	GEIECSAN	4	floating point	Accumulated CSA conv. above 16M[1]
232	E8	GEICSACE	4	binary	snapshot CSA conv. below 16M
236	EC	GEIECSAE	4	binary	snapshot CSA conv. above 16M
240	F0	GEICSAAS	4	floating point	Accumulated allocated CSA below 16M (held by the system)[1]
244	F4	GEISQAAS	4	floating point	Accumulated allocated SQA below 16M (held by the system)[1]
248	F8	GEIECSAS	4	floating point	Accumulated allocated CSA above 16M (held by the system)[1]
252	FC	GEIESQAS	4	floating point	Accumulated allocated SQA above 16M (held by the system)[1]
256	100	GEIBATCS	4	floating point	Accumulated allocated CSA below 16M (held by BATCH initiators)[1]
260	104	GEIBATEC	4	floating point	Accumulated allocated SQA below 16M (held by BATCH initiators)[1]
264	108	GEIBATSQ	4	floating point	Accumulated allocated CSA above 16M (held by BATCH initiators)[1]
268	10C	GEIBATES	4	floating point	Accumulated allocated SQA above 16M (held by BATCH initiators)[1]
272	110	GEIASCCS	4	floating point	Accumulated allocated CSA below 16M (held by ASCH initiators)[1]
276	114	GEIASCEC	4	floating point	Accumulated allocated SQA below 16M (held by ASCH initiators)[1]
280	118	GEIASCSQ	4	floating point	Accumulated allocated CSA above 16M (held by ASCH initiators)[1]
284	11C	GEIASCES	4	floating point	Accumulated allocated SQA above 16M (held by ASCH initiators)[1]
288	120	GEIOMVCS	4	floating point	Accumulated allocated CSA below 16M (held by OMVS initiators)[1]
292	124	GEIOMVEC	4	floating point	Accumulated allocated SQA below 16M (held by OMVS initiators)[1]
296	128	GEIOMVSQ	4	floating point	Accumulated allocated CSA above 16M (held by OMVS initiators)[1]

Offsets		Name	Length	Format	Description
Dec	Hex				
300	12C	GEIOMVES	4	floating point	Accumulated allocated SQA above 16M (held by OMVS initiators)[1]
304	130	GEIMTFLG	1	binary	Indicators for the current MINTIME **Bit** **Meaning When Set** **0** IPS changed during this MINTIME **1** CSA amounts incomplete in system CAUB **2** SQA amounts incomplete in system CAUB **3** Unexpected VSM error **4** System is in goal mode **5** WLM data not available for this MINTIME **6-7** Reserved
305	131	*	3	*	Reserved
308	134	GEISLID	4	EBCDIC	ID of slip trap
312	138	GEIIPLTI	8	EBCDIC	IPL time in TOD format (local time)
320	140	GEIWLMTK	8	EBCDIC	WLM token
328	148	GEISPLXI	8	EBCDIC	Sysplex name
336	150	GEISYSNM	8	EBCDIC	MVS system name
344	158	GEIMAXAS	4	binary	Maximum number of address spaces
348	15C	GEIESPMB	4	floating point	Storage frame movement count: page movement to expanded storage at begin of MINTIME
352	160	GEIESPME	4	floating point	Storage frame movement count: page movement to expanded storage at end of MINTIME
356	164	GEIESMRB	4	floating point	Storage frame movement count: migration from expanded storage to auxiliary storage at begin of MINTIME
360	168	GEIESMRE	4	floating point	Storage frame movement count: migration from expanded storage to auxiliary storage at end of MINTIME
364	16C	GEIMDL	16	EBCDIC	CPC model identifier
380	17C	GEISEQ	16	EBCDIC	CPC sequence number
396	18C	GEILOAL	4	floating point	User region value allocated below 16M[1]
400	190	GEIHIAL	4	floating point	LSQA/SWA/229/230 value allocated below 16M[1]
404	194	GEIELOAL	4	floating point	User region value allocated above 16M[1]
408	198	GEIEHIAL	4	floating point	LSQA/SWA/229/230 value allocated above 16M[1]
412	19C	GEITOTPI	4	floating point	Total number of paged-in pages, excluding swap-in, VIO, and hiperspace page-ins
416	1A0	GEISLTA	4	floating point	Number of currently available slots[1]
420	1A4	GEIRLMO	4	floating point	Number of large memory objects allocated in the system[1]

Offsets		Name	Length	Format	Description
Dec	Hex				
424	1A8	GEIRLPR	4	floating point	Number of 1 MB frames backed in real storage[1]
428	1AC	GEICMO	4	floating point	Number of 64-bit common memory objects allocated[1]
432	1B0	GEICFR	4	floating point	Number of 64-bit common memory frames backed in real storage[1]
436	1B4	GEICFFR	4	floating point	Number of 64-bit common memory frames fixed in real storage[1]
440	1B8	GEICASL	4	floating point	Number of 64-bit common memory auxiliary storage slots[1]
444	1BC	GEISMO	4	floating point	Number of shared memory objects allocated[1]
448	1C0	GEISFR	4	floating point	Number of high virtual shared memory frames backed in real storage[1]
452	1C4	GEICSIZ	8	floating point	High common area size
460	1CC	GEISSIZ	8	floating point	Shared area size[1]
468	1D4	GEILSIZ	4	floating point	Maximum large frame size[1]
472	1D8	GEIRTFIX	4	floating point	Total number of fixed frames[1]
476	1DC	GEIRBFIX	4	floating point	Number of fixed frames below 16 MB real storage[1]

[1] Sum of values obtained at each sample. To obtain average values, divide by the number of valid samples (SSHSMPNR).

ERBGGDG3 - Global gatherer data table

Offsets		Name	Length	Format	Description
Dec	Hex				
Control Flow Section					
0	0	GGDGGDG3	5	EBCDIC	Acronym 'GGDG3'
5	5	GGDRMFV	1	EBCDIC	Control block version X'08'
6	6	*	2	*	Reserved
8	8	GGDCRETR	4	binary	Pointer to RETG3 foot print area used for recovery
12	C	GGDMODPT	4	binary	Pointer to GGDMODAR area, array of all gatherer modules
16	10	GGDGOPPT	4	binary	Pointer to GGDGOPT area, gatherer options
20	14	GGDCDCBP	4	binary	Message DCB pointer
24	18	GGDTOFAG	4	binary	Total number of failures of all gatherer modules
28	1C	GGDALLPT	4	binary	Pointer to ERBMFALL module
32	20	GGDCYECB	4	binary	Cycle time ECB **Bit** **Meaning When Set** **0** Cycle time ECB is waited on **1** Cycle time ECB is posted **2-31** Reserved
36	24	GGDSMPNR	4	binary	Sample sequence number
40	28	GGDCBADS	4	binary	Number of consecutive failing samples
44	2C	GGDCBADT	4	binary	Number of consecutive failing for debugging purpose samples threshold value
48	30	GGDCFLAG	4	binary	Gatherer control flags **Bit** **Meaning When Set** **0** Gatherer initializes **1** Gatherer terminates **2** MINTIME ended **3** SMF interval ended **4** Not fully initialized because the first set-of-samples will be thrown away **5** New JES2 interface available **6-31** Reserved
52	34	GGDSTDIF	8	EBCDIC	Local Greenwich time
60	3C	GGDCTCYC	8	EBCDIC	Cycle value in TOD format
68	44	GGDCTSTP	8	EBCDIC	Stop time in TOD format
76	4C	GGDCTMNT	8	EBCDIC	MINTIME in TOD format
84	54	GGDCTCUC	8	EBCDIC	Begin current cycle in TOD format
92	5C	GGDCTNXC	8	EBCDIC	Begin next cycle TOD format
100	64	GGDCTCUS	8	EBCDIC	Begin current set-of-samples
108	6C	GGDCTNXS	8	EBCDIC	Begin next set-of-samples
Wrap-around Storage Management Section					
116	74	GGDWSHPT	4	binary	Pointer to wrap-around storage header
120	78	GGDWSHTL	4	binary	Total length of wrap-around buffer
124	7C	GGDWSHSP	4	binary	Subpool number of wrap-around buffer

ERBGGDG3 - Global gatherer table

Offsets		Name	Length	Format	Description
Dec	Hex				
Set-of-samples Section					
128	80	GGDSBEGG	8	EBCDIC	Begin time gatherer
136	88	GGDSTBEC	8	EBCDIC	Begin time current sample
144	90	GGDSTENC	8	EBCDIC	End time current sample
152	98	GGDFSSHP	4	binary	Pointer to first SSH control block
156	9C	GGDLSSHP	4	binary	Pointer to last SSH control block
160	A0	GGDCSSHP	4	binary	Pointer to current SSH control block
164	A4	GGDSSHSP	4	binary	Subpool of current set-of-sample area
ENQ Collection Data Space VIAADDR					
168	A8	GGDDSALE	4	EBCDIC	Alet of data space
172	AC	GGDDSORG	4	binary	Origin of data space
Cross-Memory Section					
176	B0	GGDXCELL	4	binary	Pointer to first cell element
180	B4	GGDXCNTR	4	binary	Counter for CDS
184	B8	GGDXETDP	4	binary	Pointer to entry table description
188	BC	GGDJXCPT	4	binary	Pointer to JXCG3 table
192	C0	GGDLXNUM	4	binary	Number of LXs requested
196	C4	GGDLXVAL	4	binary	LX value
200	C8	GGDTKNUM	4	binary	Number of ETs created
204	CC	GGDTKVAL	4	binary	Token returned by ETCRE
HSM Section					
208	D0	GGDMWELE	2	binary	Length of copied MWE part
210	D2	GGDSTALE	2	binary	Length of copied STA part
Data Set Support Section					
212	D4	GGDDSTCB	4	binary	Address of DS subtask TCB
216	D8	GGDDSNPT	4	binary	Pointer to data set names table DSNG3
220	DC	GGDDSSPT	4	binary	Pointer to data set support table DSSG3
224	E0	GGDDSSCT	4	binary	Counter of samples that should have been, but have not been, recorded on DS counted by ERB3GISS
228	E4	GGDDSECB	4	binary	DS stop ECB - DS subtask signals stop complete **Bit** **Meaning When Set** **0** DS stop ECB is waited on **1** DS stop ECB is posted **2-31** Reserved
232	E8	GGDSAVPT	4	binary	Pointer to store subchannel save area
236	EC	GGDIOSPT	4	binary	Pointer to IOSB control block
240	F0	GGDSHBPT	4	binary	Pointer to SCHIB control block
Miscellaneous Section					
244	F4	GGDPMTPT	4	binary	Pointer to performance measurement block
248	F8	GGDCPUVN	1	EBCDIC	CPU version number
249	F9	*	3	*	Reserved

Offsets		Name	Length	Format	Description
Dec	Hex				
252	FC	GGDWSIPT	4	binary	Pointer to wrap-around storage index header
256	100	GGDSID	4	EBCDIC	SMF system-id field
260	104	GGDJESJN	8	EBCDIC	JES jobname
268	10C	GGDJESAS	2	binary	JES ASID number
270	10E	*	2	*	Reserved
272	110	GGDSYNPT	4	binary	Pointer to SYNG3 table
276	114	GGDMNTPT	4	binary	Pointer to temporary OPER MOUNT area
280	118	GGDFLPCT	4	binary	FLPA/EFLPA frames, calculated at initialization
284	11C	*	4	*	Reserved
288	120	GGDASCPT	4	binary	Pointer to ASCG3 table
292	124	GGDCAPPT	4	binary	Binary of common WLM services data capsule
296	128	GGDSPIPT	4	binary	Binary of service policy chain
300	12C	GGDCSVPP	4	binary	Pointer to current SVPG3
304	130	GGDCSRQP	4	binary	Pointer to RQAA capsule
308	134	GGDBDDPT	4	binary	Pointer to diagnose x'204' data area
312	138	GGDCPUXP	4	binary	Pointer to gatherer internal CPUX3 snapshot area
316	13C	GGDCEDAA	4	binary	Pointer to enclave data area
320	140	GGDCEDCC	4	binary	Enclave data cycle count
324	144	GGDCEDFL	4	binary	Enclave data flags **Bit** **Meaning When Set** **0** Enclave data in cycle **1** Enclave data in MINTIME **2-31** Reserved
328	148	GGDDDNTP	4	binary	Pointer to temporary data set name table
332	14C	GGDBDDPG	4	binary	Length of Diagnose X'204' data in 4K-byte pages
336	150	GGDCSFLG	1	binary	Current set-of-samples control flags **Bit** **Meaning When Set** **0** Write in progress **1** Message ERB321I issued **2-31** Reserved
337	151	GGDCSUCT	3	binary	Current number of users reading the uncompressed data
340	154	GGDCSARL	4	binary	Current set-of-samples area length
344	158	GGDCSARP	4	binary	Current set-of-samples area address (in SUBSSHCF)
348	15C	GGDVRICA	4	binary	VSAMRLS data control area pointer
352	160	GGDOPDCA	4	binary	OMVS process data control area pointer
356	164	*	12	*	Reserved

GGDMODSE - Module Dependent Slot Entry Area

Offsets		Name	Length	Format	Description
Dec	Hex				
0	0	GGDMODEN (44)	48	EBCDIC	Module dependent slot

GGDMODAR - Module Dependent Area

Offsets		Name	Length	Format	Description
Dec	Hex				
0	0	GGDAUFL1	1	binary	Automatic storage control flag #1 **Bit** **Meaning When Set** **0** Storage assigned **1** Storage must not be freed **2-31** Reserved for user exit routine
1	1	GGDAUFL2	1	binary	Automatic storage control flag #2 **Bit** **Meaning When Set** **0** Area for STA getmained **1-31** Reserved
2	2	GGDAUSBP	2	binary	Subpool number
4	4	GGDAULEN	4	binary	Length of automatic area
8	8	GGDAUPTR	4	binary	Address of automatic area
12	C	GGDMODNA	8	EBCDIC	Module name
20	14	GGDMODAD	4	binary	Entry address of module
24	18	GGDBADMC	1	binary	Consecutive failures this module
25	19	GGDREDNR	1	binary	RED number index
26	1A	GGDREDID	1	binary	RED id
27	1B	GGDMODFL	1	binary	Flag bits for this module **Bit** **Meaning When Set** **0** This module selected to gatherer data **1** This module had permanent error **2** SDUMP requested, continue at retry binary **3-31** Reserved
28	1C	GGDTOFAM	4	binary	Total number of failures of this module
32	20	*	16	*	Reserved

RETG3 - Retry and Footprint Area

Offsets		Name	Length	Format	Description
Dec	Hex				
0	0	RETRETG3	5	EBCDIC	Acronym 'RETG3'
5	5	RETRMFV	1	EBCDIC	RETG3 control block version X'3'
6	6	RETSTACT	2	binary	Current stack count
8	8	RETSAVE	72	binary	Save area for ERB3GESA
80	50	*	12	*	Reserved

Offsets		Name	Length	Format	Description
Dec	Hex				
92	5C	RETFOOTP	1	binary	Footprint area **Bit** **Meaning When Set** **0** ERB3GINI entered **1** ERB3GTER entered **2** Reserved **3** ERB3GDAS entered **4** ERB3GSTO entered **5** ERB3GJS2 entered **6** ERB3GJS3 entered **7** ERB3GHSM entered
93	5D	*	1	binary	**Bit** **Meaning When Set** **0** ERB3GENQ entered **1** ERB3GMSU entered **2** ERB3GISS entered **3** ERB3GADR entered **4** ERB3GGET entered **5** ERB3GUSR entered **6** ERB3GDSI entered **7** ERB3GGSS entered
94	5E	*	1	binary	**Bit** **Meaning When Set** **0** ERB3GMES entered **1** ERB3GSMF entered **2** ERB3GSIS entered **3** ERB3GSMS entered **4** ERB3GXCF entered **5** ERB3GXCC entered **6** ERB3GMGP entered **7** ERB3GCSR entered
95	5F	*	1	binary	**Bit** **Meaning When Set** **0** ERB3GIXC entered **1** ERB3GIXI entered **2** ERB3GSTH entered **3** ERB3GCFS entered **4** ERB3GCFC entered **5** ERB3GCFI entered **6** ERB3GMRC entered **7** ERB3GMRG entered
96	60	*	1	binary	**Bit** **Meaning When Set** **0** ERB3GSCM entered **1** ERB3GRQA entered **2** ERB3GSMG entered **3** ERB3GSIG entered **4** ERB3GEN0 entered **5** ERB3GEN1 entered **6** ERB3GEN2 entered **7** ERB3GEN3 entered

Offsets		Name	Length	Format	Description
Dec	Hex				
97	61	*	1	binary	**Bit** **Meaning When Set** **0** ERB3GEN5 entered **1** ERB3GJSX entered **2** ERB3GSM2 entered **3** ERB3GHFS entered **4** ERB3GCTC entered **5** ERB3GVRI entered **6** ERB3GOPD entered **7** Reserved
98	62	*	2	*	Reserved
100	64	RETFLAG2	1	binary	Retry flag #2 **Bit** **Meaning When Set** **0** Recursion **1** XMEM established **2** ENQ environment established **3** Message data set opened **4** ERB3GXIT was entered **5** ERB425I issued for JES2 **6** ERB425I issued for JES3 **7** ERB425I issued for HSM
101	65	RETFLAG3	1	binary	Retry flag #3 **Bit** **Meaning When Set** **0** ALESERV issued for data space, set on by ERB3GENQ, set off by ERB3GTER **1-3** Reserved **4** Store subchannel entered **5-7** Reserved
102	66	RETFLAG4	1	binary	Retry flag #4 **Bit** **Meaning When Set** **0** User exit routine loaded **1** ERB3GDSI entered via error recovery module erb3gesa **2** ERB3GTEQ entered **3** ERB3GXTE entered **4** Cancel TTIMER request **5** ERBSMFI loaded **6-7** Reserved
103	67	*	29	*	Reserved
132	84	RETSTAAR (10)	96	binary	Retry stack area

RETSTACK - Retry Stack Element

Offsets		Name	Length	Format	Description
Dec	Hex				
0	0	RETTIMBE	8	EBCDIC	Time stamp begin
8	8	RETADDR	4	binary	Pointer to retry routine or zero
12	C	RETCOUNT	4	binary	Retry count for this CSECT
16	10	RETAMEMP	4	binary	Pointer to module dependent slot

Offsets		Name	Length	Format	Description
Dec	Hex				
20	14	RETFLAG1	2	binary	Retry flag #1 **Bit** **Meaning When Set** **0-1** Reserved **2** Issue message ERB280I **3** Issue message ERB268I **4** Issue message ERB269I **5-15** Reserved
22	16	RETRUBFL	2	binary	Select registers flag
24	18	RETREGSA	64	binary	Save area for RUB
88	58	RETLSIDX	4	binary	Offset of link stack entry to recover
92	5C	*	4	*	Reserved

GGDGOPT - Options Area

Offsets		Name	Length	Format	Description
Dec	Hex				
0	0	GGDGOCYC	4	binary	Gatherer option CYCLE
4	4	GGDGOSTP	4	binary	Gatherer option STOP **Bit** **Meaning When Set** **0** NOSTOP option active **1-31** Reserved
8	8	GGDGOSYN	4	binary	Gatherer option SYNCH **Bit** **Meaning When Set** **0** NOSYNCH option active **1-31** Reserved
12	C	GGDGOMNT	4	binary	Gatherer option MINTIME
16	10	*	3	*	Reserved
19	13	GGDGOCLA	1	EBCDIC	Gatherer option SYSOUT: sysout class alphanumeric value
20	14	GGDGORES	2	binary	Resource selected options **Bit** **Meaning When Set** **0** Resource Processor selected **1** Resource DASD selected **2** Resource Storage selected **3** Resource JES2 selected **4** Resource JES3 selected **5** Resource HSM selected **6** Resource ENQ selected **7** Resource Oper MSG selected **8** Resource Oper MOUNT selected **9** Resource XCF selected **10** Resource OMVS selected **11-14** Reserved **15** Resource User selected

ERBGGDG3 - Global gatherer table

Offsets		Name	Length	Format	Description
Dec	Hex				
22	16	GGDGOFL1	1	binary	Flag byte #1 **Bit** **Meaning When Set** **0** Performance measurement active **1** Data set support selected **2** IPM measurement requested **3** CFDETAIL requested **4** CACHE requested **5** VSAMRLS requested **6** OPD requested **7** Reserved
23	17	*	1	*	Reserved
24	18	GGDJESN	4	EBCDIC	Name of JES subsystem chosen
28	1C	GGDGOWHL	4	binary	Value data set option WHOLD
32	20	GGDGOWST	4	binary	Value of option WSTOR
36	24	GGDGOSOF	4	binary	Offset of synch point from the full hour used by gatherer. Units are full seconds.
40	28	GGDGOCCU	4	binary	Pointer to bit array for selected cache SSIDs
44	2C	*	32	*	Reserved

ERBOPDG3 - OMVS process data table

Offsets		Name	Length	Format	Description
Dec	Hex				
OPDG3 Header Section:					
0	0	OPDOPDG3	5	EBCDIC	Acronym 'OPDG3'
5	5	OPDVERG3	1	EBCDIC	Control block version X'01'
6	6	OPDHDRLE	2	binary	Length of OPDG3 header
8	8	OPDTOTLE	4	binary	Total length of OPDG3
12	C	OPDENTO	4	binary	Offset to OPDG3 array
16	10	OPDENTL	4	binary	Length of OPDG3 entry
20	14	OPDENTN	4	binary	Number of OPDG3 entries
24	18	OPDSUMO	4	binary	Offset to Summary data (OSDG3)
28	1C	OPDSUML	4	binary	Summary data length
32	20	OPDSTAT	4	binary	Flags: **Bit** **Meaning When Set** **0** OMVS inactive **1-31** Reserved
36	24	*	92	*	Reserved
OSDG3: Summary information					
0	0	OSDPROC	8	EBCDIC	OMVS procedure name
8	8	*	2	*	Reserved
10	A	OSDKASID	2	binary	Kernel address space ID
12	C	OSDPLIST	40	EBCDIC	OMVS parmlib member list
OPDG3 Array Entry:					
0	0	OPDJOBNM	8	EBCDIC	Job name (as noted in ASCB)
8	8	OPDUSER	8	EBCDIC	User name (from login)
16	10	OPDPID	4	binary	Process ID
20	14	OPDPPID	4	binary	Parent's process ID
24	18	OPDASID	2	binary	Address space ID. Undefined state if 0.
26	1A	*	5	*	Reserved
31	1F	OPDSTYY	4	EBCDIC	4-digit year
35	23	OPDSTDD	3	EBCDIC	3-digit day of year (1-366)
38	26	OPDSTHH	2	EBCDIC	Process start time hour
40	28	OPDSTMM	2	EBCDIC	Process start time minute
42	2A	OPDSTSS	2	EBCDIC	Process start time second
44	2C	OPDCT	8	EBCDIC	Process system and user compute time in STCK format
52	34	OPDSTAT1	1	binary	MVS status flags: **Bit** **Meaning When Set** **0** Space swapped out **1** Ptrace kernel wait **2-7** Reserved

Offsets		Name	Length	Format	Description
Dec	Hex				
53	35	OPDSTAT2	1	binary	Process status flags: **Bit** **Meaning When Set** **0** Process stopped **1** Reserved **2** multiple threads **3** pthread task in process **4-7** Reserved
54	36	OPDSTAT3	1	EBCDIC	State of reported task: **A** Message queue receive wait **B** Message queue sent wait **C** Communication system kernel wait **D** Semaphore operation wait **E** Quiesce frozen **F** File system kernel wait **G** MVS pause wait **H** Multiple threads, pthread_create used **I** Swapped out **K** Other kernel wait **L** Cancelled, parent waits **M** Multiple threads, no pthread_create used **P** Ptrace kernel wait **Q** Quiesce termination wait **R** Running **S** Sleeping **W** Waiting for child **X** Creating new process **Z** Zombie. Cancelled, Parent does not wait
55	37	*	1	*	Reserved
56	38	OPDLWPID	4	binary	Latch process ID the process is waiting for (0 = not waiting)
60	3C	OPDGFLGS	4	binary	General flags: **Bit** **Meaning When Set** **0** Server information is valid (in fields OPDSNAME, OPDAFILE, OPDMFILE, OPDSTYPE) **1-31** Reserved
64	40	OPDSNAME	32	EBCDIC	Server name in mixed case
96	60	OPDAFILE	4	binary	Number of active files
100	64	OPDMFILE	4	binary	Maximum number of files
104	68	OPDSTYPE	4	binary	Server type
108	6C	OPDCMND	40	EBCDIC	Truncated command buffer in mixed case
148	94	OPDDCT	8	EBCDIC	Delta TCB time

ERBRCDG3 - Resource collection data

Resource Collection Data Header

Offsets Dec	Offsets Hex	Name	Length	Format	Description
0	0	RCDACRO	5	EBCDIC	Acronym 'RCDG3'
5	5	RCDVERS	1	binary	Control block version X'08'
6	6	RCDHLEN	2	binary	Size of RCDHDR
8	8	RCDSIZ	4	binary	Size of all resource collection data. This includes RCDHDR, RCDBMAP, RCDG3, RCDPD, RCDRD and RCDSD.
12	C	RCDPNAM	8	EBCDIC	Policy name
20	14	RCDPTM	8	binary	Local time policy was activated (TOD format)
28	1C	RCDNTVL	4	binary	Current sample interval (in milliseconds). This is the frequency with which WLM samples delays reported in the RCAA.
32	20	RCDNTV#	4	binary	Total number of times WLM sampling code ran. A monitor issuing successive calls to IWMRCOLL should not assume that WLM sampling code ran at the interval specified by RCDNTVL between its calls. This field can be used to translate sampled state data into actual percentages of time.
36	24	RCDMSC#	2	binary	Maximum possible number of service classes according to SVPOL service class array
38	26	RCDMRC#	2	binary	Maximum possible number of report classes according to SVPOL report class array
40	28	RCDMPD#	2	binary	Maximum possible number of service or report class period entries according to SVPOL.
42	2A	RCDMRD#	2	binary	Maximum possible number of response time distribution buckets according to number of periods with response time goals
44	2C	RCDBMPL	2	binary	Length of an entry in the response time distribution mapping array
46	2E	RCDBMP#	2	binary	Number of response time distribution buckets
48	30	RCDBMPO	4	binary	Offset from begin of RCDHDR to response time distribution mapping array (RCDBMAP)
52	34	RCDSCAL	2	binary	Length of one RCDG3 workload activity entry in the RCDSCOF array
54	36	RCDSCA#	2	binary	Number of entries in RCDSCOF array. This is the number of service classes returned in IWMSVPOL by IWMPQRY.
56	38	RCDSCOF	4	binary	Offset from begin of RCDHDR to array of RCDG3 entries. These entries represent service classes.
60	3C	RCDRCAL	2	binary	Length of one RCDG3 workload activity entry in the RCDRCOF array
62	3E	RCDRCA#	2	binary	Number of entries in RCDRCOF array. This field is the number of report classes returned in IWMSVPOL by IWMPQRY.
64	40	RCDRCOF	4	binary	Offset from begin of RCDHDR to array of RCDG3 entries

Offsets		Name	Length	Format	Description
Dec	Hex				
68	44	RCDPDAL	2	binary	Length of one RCDG3 period entry in the RCDPD array
70	46	RCDPDA#	2	binary	Number of entries in the RCDPD array
72	48	RCDPDAO	4	binary	Offset from begin of RCDHDR to begin of RCDPD array
76	4C	RCDRDAL	2	binary	Length of one RCDG3 response time bucket entry in the RCDRD array
78	4E	RCDRDA#	2	binary	Number of entries in the RCDRD array
80	50	RCDRDAO	4	binary	Offset from begin RCDHDR to begin of RCDRD array
84	54	RCDSDAL	2	binary	Length of one RCDG3 subsystem delay data entry in the RCDSD array
86	56	RCDSDA#	2	binary	Number of entries in the RCDSD array
88	58	RCDSDAO	4	binary	Offset from begin of RCDHDR to begin of RCDSD array
92	5C	RCDSUBP	2	binary	Subsystem phase count X'0002'
94	5E	*	2	*	Reserved
96	60	RCDMADJ	4	binary	Value of RMCTADJC - adjustment factor for CPU rate
100	64	RCDNFFI	4	binary	Normalization factor for zAAP. Multiply zAAP times or service units with this value and divide by 256 to calculate the CP equivalent value.
104	68	RCDNFFS	4	binary	Normalization factor for zIIP. Multiply zIIP service units with this value and divide by 256 to calculate the CP equivalent value.
Response Time Distribution Map Array					
0	0	RCDBENT	4	binary	Response time distribution bucket mappings. Each word defines a maximum % of a goal (ie. 50, 70, 100, etc.) When used in conjunction with an RCDDENT, a monitor product can show the number of transactions that completed in a percentage of a goal. The last entry in the array contains X'FFFFFFFF'. This indicates that this bucket includes all transactions that completed with longer response times than the previous bucket.
Resource Collection Data Entry					
0	0	RCDTYPE	1	binary	What this RCDG3 entry represents **Bit** **Meaning When Set** **0** Service class **1** Report class **2-7** Reserved
1	1	RCDFLGS	1	binary	Class data availability flags **Bit** **Meaning When Set** **0** Service classes served **1-7** Reserved

Offsets		Name	Length	Format	Description
Dec	Hex				
2	2	RCDCLX	2	binary	Index into the service class or report class list mapped by SVPCD and SVPHD, respectively (service policy information)
4	4	RCDMP#	1	EBCDIC	Maximum possible number of periods for this RCDG3.
5	5	RCDMB#	1	EBCDIC	Maximum possible number of response time distribution buckets for this RCDG3.
6	6	RCDPD#	2	binary	Number of period data entries for this RCDG3 entry
8	8	RCDPDI	4	binary	Index into RCDG3 period entry array
12	C	RCDFRX	2	binary	Index to first RT-distribution bucket of this class
14	E	RCDCR#	2	binary	Number of buckets for this class
16	10	RCDFSX	2	binary	Index to first subsystem delay data entry of this class
18	12	RCDCS#	2	binary	Number of subsystem delay data entries for this class
Resource Collection Data - Period Entry					
0	0	RCDPFLGS	1	binary	Data availability flags **Bit** **Meaning When Set** **0** Resource consumption data **1** Response time data **2** General execution delay data **3-7** Reserved
1	1	RCDPFLG1	1	binary	Report class period flags **Bit** **Meaning When Set** **0** Heterogeneous report class period **1-7** Reserved
2	2	RCDPLSC	2	binary	Index of the service class that last contributed to this report class. For homogeneous report class periods, this service class period's goal has to be used to format the response time distribution for ended transactions reported in this report class. Zero for a service class entry.
4	4	RCDPERI	1	binary	Period number
5	5	RCDRD#	1	binary	Number of entries in the response time distribution bucket array (RCDRD) that belong to this period or zero
6	6	RCDRDI	2	binary	Index into response time distribution bucket array. This field will be zero when there are no response time goals specified.
8	8	RCDSD#	2	binary	Number of entries in the subsystem work manager delay array (RCDSD) that belong to this period or zero
10	A	RCDSDI	2	binary	Index into subsystem work manager delay data array. Zero means, there is no subsystem work manager delay data for this period.
12	C	RCDCPU	8	binary	Total CPU service units for this period
20	14	RCDSRB	8	binary	Total SRB service units for this period

Offsets		Name	Length	Format	Description
Dec	Hex				
28	1C	RCDRCP	4	binary	Count of transaction completions for this period. This field also includes transaction completions reported by subsystem work managers via the IWMRPT service.
32	20	RCDARCP	4	binary	Count of transactions that completed abnormally as reported by subsystem work managers. This value is not part of RCDRCP and should not be used for response time calculations.
36	24	RCDNCP	4	binary	Count of transactions that completed their execution phase as reported by subsystem work managers via the IWMMNTFY service.
40	28	RCDANCP	4	binary	Count of transactions that completed their execution phase abnormally as reported by subsystem work manager. This value is not part of RCANCP and should not be used for execution response time calculations.
44	2C	RCDTET	8	binary	Total transaction elapsed time (in 1024-microsecond units)
52	34	RCDXET	8	binary	Total transaction execution time (in 1024-microsecond units)
60	3C	RCDCUSE	4	binary	Total using samples
64	40	RCDTOTD	4	binary	Total delay samples used in SRM's execution velocity calculation
68	44	RCDQDT	8	binary	Queue delay time (in 1024-microsecond units)
76	4C	RCDADT	8	binary	Resource affinity delay time (in 1024-microsecond units)
84	54	RCDCVT	8	binary	JCL conversion delay time (in 1024-microsecond units)
92	5C	RCDIQT	8	binary	Ineligible queue time (in 1024-microsecond units)
100	64	RCDRCT	4	binary	Total region control task time in microsecond units
104	68	RCDIIT	4	binary	Total I/O interrupt time in microsecond units
108	6C	RCDHST	4	binary	Total hiperspace service time in microsecond units
112	70	RCDIFAT	8	binary	Total zAAP service time in microsecond units. Multiply with RCDNFFI and divide by 256 to calculate the equivalent time on a CP.
120	78	RCDIFCT	8	binary	Total zAAP service time spent on CPs in microsecond units
128	80	RCDIFASU	8	binary	Total zAAP service units. Multiply with RCDNFFI and divide by 256 to calculate the CP equivalent value
136	88	RCDIFASUCP	8	binary	Total zAAP service units spent on CPs
144	90	RCDSUPSU	8	binary	Total zIIP service units. Multiply with RCDNFFS and divide by 256 to calculate the CP equivalent value
152	98	RCDSUPSUCP	8	binary	Total zIIP service units spent on CPs

Offsets		Name	Length	Format	Description
Dec	Hex				
160	A0	RCDTPDP	8	binary	Total CPU time spent for work units with promoted dispatching priority (in 1024-microsecond units).
168	A8	RCDCPUDL	4	binary	CP delay samples
172	AC	RCDAAPDL	4	binary	zAAP delay samples
176	B0	RCDIIPDL	4	binary	zIIP delay samples
180	B4	RCDRGCAP	4	binary	Resource group capping delay samples
Resource Collection Data - Response Time Distribution Array					
0	0	RCDDENT	4	binary	An entry in the RCDG3 response time distribution array. Each entry in the array contains the number of transactions that completed in the time period represented by that entry. When used with the response time distribution bucket mapping (RCDBMAP), monitors can construct a distribution of completions versus goals specified.
Resource Collection Data - Subsystem Work Manager Delays					
0	0	RCDSTYP	4	EBCDIC	Subsystem type, as used in the classification rules specified in the WLM administrative application
4	4	RCDEFLG	1	binary	Flags **Bit** **Meaning When Set** **0** Represents states sampled in the begin-to-end phase of a transaction **1** Represents states sampled in the execution phase of a transaction **2-7** Reserved
5	5	*	3	*	Reserved
8	8	RCDESS#	4	binary	Total number of transaction states sampled in the work phase specified by RCDEFLG
12	C	RCDACTV	4	binary	Total number of active state samples. Active indicates that there is a program executing on behalf of the work request, from the perspective of the work manager. This does not mean that the program is active from the base control program's perspective.
16	10	RCDRDY	4	binary	Total number of ready state samples. Ready indicates that there is a program ready to execute on behalf of the work request described by the monitoring environment, but the work manager has given priority to another work request.
20	14	RCDIDL	4	binary	Total number of idle state samples. Idle indicates that no work request is available to the work manager that is allowed to run.
24	18	RCDWLOK	4	binary	Total number of waiting for lock state samples
28	1C	RCDWIO	4	binary	Total number of waiting for I/O state samples. Waiting for I/O indicates that the work manager is waiting for an activity related to an I/O request. This may be an actual I/O operation or some other function associated with the I/O request.

Offsets		Name	Length	Format	Description
Dec	Hex				
32	20	RCDWCON	4	binary	Total number of waiting for conversation state samples. Waiting for conversation may have been used in conjunction with the WLM service IWMMSWCH to identify where the recipient of the conversation is located. In this case, only the switched state will be recorded.
36	24	RCDWDST	4	binary	Total number of waiting for distributed request state samples. Waiting for distributed request indicates a high level that some function or data must be routed prior to resumption of the work request. This is to be contrasted with waiting for conversation, which is a low level view of the precise resource that is needed. A distributed request could involve waiting on a conversation as part of its processing.
40	28	RCDWSL	4	binary	Waiting for a session to be established locally, ie. on the current MVS image
44	2C	RCDWSN	4	binary	Waiting for a session to be established somewhere in the network
48	30	RCDWSS	4	binary	Waiting for a session to be established somewhere in the sysplex
52	34	RCDWTMR	4	binary	Waiting for a timer
56	38	RCDWO	4	binary	Waiting for another product
60	3C	RCDWMSC	4	binary	Waiting for unidentified resource, possibly among another more specific category, but which may not be readily determined
64	40	RCDSSL	4	binary	State representing transactions for which there are logical continuations on this MVS image. Subsystem work managers might set this state when they function ship a transaction to another component within the same MVS image.
68	44	RCDSSS	4	binary	State representing transactions for which there are logical continuations on another MVS image in the sysplex. Subsystem work managers might set this state when they function ship a transaction to another component on another MVS image within the sysplex.
72	48	RCDSSN	4	binary	State representing transactions for which there are logical continuations somewhere within the network. Subsystem work managers might set this state when they function ship a transaction to another component within the network.
76	4C	RCDBPMI	4	binary	Number of state samples representing DB2 buffer pool misses that resulted in I/O.
80	50	*	12	*	Reserved
92	5C	RCDWNL	4	binary	Total number of state samples reflecting waiting for new latch

Offsets		Name	Length	Format	Description
Dec	Hex				
96	60	RCDACTA	4	binary	Total number of active application state samples. Active application indicates a program is executing on behalf of the work request, from the perspective of the work manager. This does not mean that the program is active from the base control program's perspective.
100	64	RCDWSSL	4	binary	Total number of waiting for an SSL thread samples
104	68	RCDWRET	4	binary	Total number of waiting for a regular thread samples
108	6C	RCDWREW	4	binary	Total number of waiting for a registration to a work table samples
112	70	RCDWTY1	4	binary	Total number of waiting for resource type 1 samples
116	74	RCDWTY2	4	binary	Total number of waiting for resource type 2 samples
120	78	RCDWTY3	4	binary	Total number of waiting for resource type 3 samples
124	7C	RCDWTY4	4	binary	Total number of waiting for resource type 4 samples
128	80	RCDWTY5	4	binary	Total number of waiting for resource type 5 samples
132	84	*	40	*	Reserved

ERBREDG3 - Resource data record

Offsets		Name	Length	Format	Description
Dec	Hex				
0	0	REDENTRY(10)	12	EBCDIC	RED Array **Entry** **Resource** **1** USER **2** PROCESSOR **3** DEVICE **4** STORAGE **5** JES2/JES3 **6** HSM **7** ENQ **8** MOUNT **9** MESSAGE **10** XCF
REDENTRY Section					
0	0	REDREDID	1	binary	Resource Data Record ID
1	1	REDFLAG1	1	binary	Flags **Bit** **Meaning When Set** **0** This resource is invalid **1** USE records available **2** WAIT records available **3-7** Reserved
2	2	*	2	*	Reserved
4	4	REDFUWDO	4	binary	Offset to first USE/WAIT record
8	8	REDUWDLE	2	binary	For all resources except ENQ: Length of USE/WAI record
8	8	REDUWDL1	1	binary	Short length of ENQ UWD record (without System/Jobname)
9	9	REDUWDL2	1	binary	Total length of ENQ UWD record (with System/Jobname)
10	A	REDUSERN	2	binary	Number of user-exit records

ERBSHDG3 - Sample header

Offsets		Name	Length	Format	Description
Dec	Hex				
0	0	SHDSHDG3	5	EBCDIC	Acronym 'SHDG3'
5	5	SHDRMFV	1	binary	Control block version X'02'
6	6	SHDLEN	1	binary	Length of SHDG3
7	7	SHDFLAG1	1	binary	Sample flag **Bit** **Meaning When Set** **0** Sample is invalid **1-7** Reserved
8	8	SHDPREVO	4	binary	Offset to previous sample. This field contains the offset within the Monitor III data gatherer areas. The Monitor III reporter module changes the offset to a pointer after the data have been moved to the reporter's address space
12	C	SHDNEXTO	4	binary	Offset to next sample. This field contains the offset within the Monitor III data gatherer areas. The Monitor III reporter module changes the offset to a pointer after the data have been moved to the reporter's address space
16	10	SHDREDOF	4	binary	Offset to first RED record
20	14	SHDREDNR	2	binary	Number of RED records
22	16	SHDREDLE	2	binary	Length of one REDG3 entry
24	18	*	6	*	Reserved
30	1E	SHDUWDNR	2	binary	Number of Use/Wait records
32	20	*	16	*	Reserved

ERBSPGG3 - Storage group and volume data

Offsets		Name	Length	Format	Description
Dec	Hex				
SPGG3 Header Section:					
0	0	SPGACR	5	EBCDIC	Acronym 'SPGG3'
5	5	SPGVER	1	binary	Control block version X'01'
6	6	*	2	*	Reserved
8	8	SPGHDRL	4	binary	Length of SPGG3 header
12	C	SPGTOTL	4	binary	Total length of SPGG3
16	10	SPGSGDATL	4	binary	Length of one storage group entry
20	14	SPGSGDATN	4	binary	Number of storage group entries
24	18	SPGSGDATO	4	binary	Offset to storage group entries
28	1C	SPGVOLDATL	4	binary	Length of one volume data entry
32	20	SPGVOLDATN	4	binary	Number of volume data entries
36	24	SPGVOLDATO	4	binary	Offset to volume data entries
40	28	SPGSTAT	2	binary	Status flags **Bit** **Meaning When Set** **0** No SPG data collected **1** Internal problem **2** SMS inactive **3-7** Reserved **8** No volume data available **9** no storage group data **10-15** Reserved
42	2A	*	6	*	Reserved
Storage Group Entry					
0	0	GNAMEL	2	binary	Actual length of storage group name
2	2	GNAME	30	EBCDIC	Storage group name
32	20	FIRSTVOL	2	binary	Index of first volume entry for this storage group
34	22	NUMBERVOL	2	binary	Number of volume entries for this storage group
36	24	*	4	*	Reserved
Volume Data Entry					
0	0	VNAMEL	2	binary	Actual length of volume name
2	2	VNAME	6	EBCDIC	Volume name (volser)
8	8	TOTALSPACE	4	binary	Total space on volume (megabyte)
12	C	FREESPACE	4	binary	Free space on volume (megabyte)
16	10	LBLOCKSIZE	4	binary	Largest block of unallocated space (megabyte)
20	14	*	4	*	Reserved

ERBSSHG3 - MINTIME set of samples header

Offsets		Name	Length	Format	Description
Dec	Hex				
SSHG3 Header Section:					
0	0	SSHSSHG3	5	EBCDIC	Acronym 'SSHG3'
5	5	SSHRMFV	1	binary	Control block version X'0D'
6	6	SSHLEN	2	binary	Length of SSHG3 header
8	8	SSHRMFVN	3	EBCDIC	RMF version number
11	B	SSHFLAG1	1	binary	Flag byte **Bit** **Meaning** **0** Data are compressed **1** WLM goal mode data **2-7** Reserved
12	C	*	24	*	Reserved
36	24	SSHSHDFO	4	binary	Offset of first sample header from ERBSSHG3
40	28	SSHSHDLO	4	binary	Offset of last sample header from ERBSSHG3
44	2C	SSHTOTLE	4	binary	Total length for this set of samples (including the set of samples header)
48	30	*	8	*	Reserved
56	38	SSHSMPNR	4	binary	Number of valid samples
60	3C	SSHTIBEG	8	binary	Begin time for this set of samples
68	44	SSHTIEND	8	binary	End time for this set of samples
76	4C	*	16	*	Reserved
92	5C	SSHASIO	4	binary	Offset to ASID table from ERBSSHG3
96	60	*	12	*	Reserved
108	6C	SSHDVTO	4	binary	Offset to DVT table from ERBSSHG3
112	70	*	8	*	Reserved
120	78	SSHENTO	4	binary	Offset to ENT table from ERBSSHG3
124	7C	*	24	*	Reserved
148	94	SSHGEIO	4	binary	Offset to GEIG3 table from ERBSSHG3
152	98	SSHIOML	1	binary	Processor type on which data was created **Value** **Meaning** **X'03'** 9672, zSeries
153	99	SSHEFLAG	1	binary	Extended storage indicators **Bit** **Meaning When Set** **0** Extended storage installed **1-7** Reserved
154	9A	SSHPRFGS	1	binary	Processor flags **Bit** **Meaning When Set** **0** ES/Connection Channel enabled **1** ES/Connection Director configured **2-7** Reserved
155	9B	*	1	*	Reserved

Offsets		Name	Length	Format	Description
Dec	Hex				
156	9C	SSHGOCYC	4	binary	Gatherer CYCLE option
160	A0	SSHGOSTP	4	binary	Gatherer STOP option. (If the first bit is set to 0, NOSTOP is in effect.)
164	A4	SSHGOSYN	4	binary	Gatherer SYNC option. (If the first bit is set to 0, NOSYNC is in effect.)
168	A8	SSHGOMNT	4	binary	Gatherer MINTIME option
172	AC	*	3	*	Reserved
175	AF	SSHGOCLA	1	EBCDIC	Gatherer SYSOUT class option
176	B0	*	4	*	Reserved
180	B4	SSHJESN	4	EBCDIC	Name of JES subsystem
184	B8	SSHGOWHL	4	binary	Gatherer DATASET WHOLD suboption
188	BC	SSHGOWST	4	binary	Gatherer WSTOR option
192	C0	*	40	*	Reserved
232	E8	SSHSTDIF	8	binary	Difference between local time and Greenwich Mean Time where the difference equals local time minus Greenwich Mean Time
240	F0	SSHHSMJN	8	EBCDIC	Jobname of HSM subsystem
248	F8	SSHHSMAS	2	binary	ASID number of HSM subsystem
250	FA	SSHJESJN	8	EBCDIC	Jobname of JES subsystem
258	102	SSHJESAS	2	binary	ASID number of JES subsystem
260	104	*	8	*	Reserved
268	10C	SSHCSRO	4	binary	Offset to CSR table from ERBSSHG3. This field contains the offset when the data are within the wrap around buffer.
272	110	SSHJLCYC	4	binary	Time-offset when the last cycle was gathered, expressed in CYCLE time units.
276	114	*	4	*	Reserved
280	118	SSHRCDO	4	binary	Offset to RCDG3 table from ERBSSHG3
284	11C	SSHCPUO	4	binary	Offset to CPUG3 table from ERBSSHG3
288	120	SSHIPLTI	8	binary	IPL time in TOD format
296	128	SSHWLMTK	8	binary	WLM token
304	130	SSHENCO	4	binary	Offset to ENCG3 table from ERBSSHG3
308	134	*	8	*	Reserved
316	13C	SSHCFIO	4	binary	Offset to CFIG3 table from ERBSSHG3
320	140	SSHCATO	4	binary	Offset to CATG3 table from ERBSSHG3
324	144	*	4	*	Reserved
328	148	SSHOPDO	4	binary	Offset to OPDG3 table from ERBSSHG3
332	14C	*	4	*	Reserved
336	150	SSHSPGO	4	binary	Offset to SPGG3 table from ERBSSHG3

Offsets		Name	Length	Format	Description
Dec	Hex				
340	154	SSHCPDO	4	binary	Offset to CPDG3 table from ERBSSHG3
344	158	*	24	*	Reserved

ERBSVPG3 - Service policy

Offsets		Name	Length	Format	Description
Dec	Hex				
SVPG3 Header Section:					
0	0	SVPNAM	5	EBCDIC	Acronym 'SVPG3'
5	5	SVPDVN	1	binary	Control block version X'01'
6	6	SVPDIL	2	binary	Length of SVPG3 header
8	8	SVPDLE	4	binary	Total length of the active service policy data structure
12	C	SVPTIB	8	EBCDIC	Begin time in TOD. Time of policy activation
20	14	SVPTIE	8	EBCDIC	End time in TOD. Time of policy deactivation
28	1C	SVPDPO	4	binary	Offset to the service policy definition section
32	20	SVPDPL	2	binary	Length of the policy entry in the policy section
34	22	*	2	*	Reserved
36	24	SVPDWO	4	binary	Offset to the workload definition section
40	28	SVPDWC	2	binary	Number of workload entries in the workload definition section
42	2A	SVPDWL	2	binary	Length of each workload entry
44	2C	SVPDCO	4	binary	Offset to the service class definition section
48	30	SVPDCC	2	binary	Number of service class entries in the service class definition section
50	32	SVPDCL	2	binary	Length of each service class definition entry
52	34	SVPDZO	4	binary	Offset of service class period entries
56	38	SVPDZC	2	binary	Number of service class periods
58	3A	SVPDZL	2	binary	Length of each service class period entry
60	3C	SVPDRO	4	binary	Offset to the report class definition section
64	40	SVPDRC	2	binary	Number of report class entries in the report class definition section
66	42	SVPDRL	2	binary	Length of each report class definition entry
68	44	SVPDGO	4	binary	Offset to the resource group definition section
72	48	SVPDGC	2	binary	Number of resource group entries in the resource group definition
74	4A	SVPDGL	2	binary	Length of each resource group definition entry
76	4C	*	52	*	Reserved
Service Policy					
0	0	SVPNSP	8	EBCDIC	Service policy name
8	8	SVPDSP	32	EBCDIC	Service policy description
40	28	SVPTPA	8	EBCDIC	Time/date (TOD format) of policy activation
48	30	SVPIPU	8	EBCDIC	Userid of the system operator or service administrator who activated the service policy
56	38	SVPSNA	8	EBCDIC	Name of the system on which policy activation was initiated
64	40	SVPSEQ	4	binary	Classification sequence number

Offsets		Name	Length	Format	Description
Dec	Hex				
68	44	SVPASN	4	binary	Activation sequence number
72	48	SVPIDN	8	EBCDIC	Name of the service definition from which the service policy was extracted
80	50	SVPTDI	8	EBCDIC	Time/date (TOD format) that the service definition was installed
88	58	SVPIDU	8	EBCDIC	Userid of the service administrator who installed the service definition
96	60	SVPIDS	8	EBCDIC	Name of the system on which the service definition was installed
104	68	SVPIDD	32	EBCDIC	Description of service definition from which the service policy was extracted
136	88	SVPCPU	4	binary	CPU service coefficient *10000 - the number by which accumulated CPU service units will be multiplied (weighted)
140	8C	SVPIOC	4	binary	I/O service coefficient * 10000 - the number by which accumulated I/O service units will be multiplied (weighted)
144	90	SVPMSO	4	binary	Storage service coefficient (MSO) * 10000 - the number by which accumulated storage service units will be multiplied (weighted)
148	94	SVPSRB	4	binary	SRB service coefficient * 10000 - the number by which accumulated SRB service units will be multiplied (weighted)
152	98	SVPECP	4	EBCDIC	EBCDIC representation of CPU service coefficient
156	9C	SVPEIO	4	EBCDIC	EBCDIC representation of I/O service coefficient
160	A0	SVPEMS	8	EBCDIC	EBCDIC representation of Storage service coefficient
168	A8	SVPESR	4	EBCDIC	EBCDIC representation of SRB service coefficient
172	AC	*	4	*	Reserved
Workload Information					
0	0	SVPWNM	8	EBCDIC	Workload name
8	8	SVPWDE	32	EBCDIC	Workload description
Service Class Information					
0	0	SVPCNM	8	EBCDIC	Service class name
8	8	SVPCDE	32	EBCDIC	Service class description
40	28	SVPCWN	8	EBCDIC	Name of the workload this service class is associated with
48	30	SVPCRN	8	EBCDIC	Name of the resource group this service class is associated with - blanks if no resource group association
56	38	SVPCPO	4	binary	Offset of service class period entries for this service class
60	3C	SVPCPN	2	binary	Number of service class periods for this service class
62	3E	*	2	*	Reserved

Offsets		Name	Length	Format	Description
Dec	Hex				
64	40	SVPCGI	4	binary	Resource group index - the index of the resource group entry in SVPRG of the resource group to which this service class belongs
68	44	SVPCWI	4	binary	Workload index - the index of the workload entry in SVPWD of the workload to which this service class belongs
72	48	SVPCRC	4	binary	Number of periods with response time goals specified
Service Class Period Information					
0	0	SVPTYP	4	binary	Goal type indicators **Bit** **Meaning When Set** **0** Percentile response time goal **1** Average response time goal **2** Velocity goal **3** Discretionary goal **4** System goal **5-7** Reserved
4	4	*	1	*	Reserved
5	5	SVPRTU	1	binary	Response time unit indicator indicating the units in which SVPVAL is expressed
6	6	SVPPER	2	binary	Goal percentile value
8	8	SVPIMP	2	binary	Importance level ranging from 1 to 5 where 1 is most important
10	A	*	2	*	Reserved
12	C	SVPVAL	4	binary	Response time goal or velocity goal. Zero if discretionary or system goal or no goal defined.
16	10	SVPDUR	4	binary	Service class period duration in service units, or zero for last period
Resource Group Information					
0	0	SVPGNM	8	EBCDIC	Resource group name
8	8	SVPGDE	32	EBCDIC	Resource group description
40	28	SVPGMN	4	binary	If bit 1 of SVPGLT is ON, this field contains the minimum capacity of the resource group in unweighted CPU service units per second. In addition, the scope of the resource group is sysplex-wide. See also the description of bit 3 and bit 4 of SVPGLT.
44	2C	SVPGMX	4	binary	If bit 0 of SVPGLT is ON, this field contains the maximum capacity of the resource group in unweighted CPU service units per second. In addition, the scope of the resource group is sysplex-wide. See also the description of bit 3 and bit 4 of SVPGLT.

Offsets		Name	Length	Format	Description
Dec	Hex				
48	30	SVPGLT	4	binary	Indicators **Bit** **Meaning When Set** **0** Maximum capacity was specified **1** Minimum capacity was specified **2** Reserved **3** Specification of SVPGMN and SVPGMX is in % of the LPAR share. The scope of the resource group is system-wide rather than sysplex-wide. **4** Specification of SVPGMN and SVPGMX is in % of a single processor (CP) capacity. The scope of the resource group is system-wide rather than sysplex-wide. **5-7** Reserved.
Report Class Information					
0	0	SVPRNM	8	EBCDIC	Report class name
8	8	SVPRDE	32	EBCDIC	Report class description

ERBUWDG3 - USE/WAIT record

Offsets		Name	Length	Format	Description
Dec	Hex				
0	0	UWDUWRID	1	binary	USE/WAIT record id **Bit** **Meaning When Set** **0** WAIT record **1** USE record **2-7** Resource identification
1	1	UWDASID	2	binary	Address space (ASIG3) table index
Extended Data for PROC Section (See resource id in UWDUWRID):					
3	3	UWDFLAGP	1	binary	Flag for processor delay types **Bit** **Meaning When Set** **0** Processor was used by enclaves **1** Processor was a zAAP **2** Processor was a standard processor used by zAAP work **3** Processor was a standard processor **4** Processor was a zIIP **5** Processor was a standard processor used by zIIP work **6-7** Reserved
Extended Data for DEV Section (See resource id in UWDUWRID):					
3	3	UWDDEVNR	2	binary	Device table (DVTG3) index
Extended Data for STOR Section See resource id in UWDUWRID):					
3	3	UWDPDEVR	2	binary	Paging device DVTG3 index
5	5	UWDFLAGS	1	binary	Flag for storage status **Bit** **Meaning When Set** **0** Delayed for LOCAL request **1** Delayed for SWAP IN request **2** Delayed for COMMON request **3** Delayed for VIO request **4** Space type LOCL **5** Reserved **6** Space type COMM **7** Space type PLPA
Extended Data for JES2/JES3 section (See resource id in UWDUWRID):					
3	3	UWDJESFU	2	binary	JES2/JES3 function code For a list of JES function codes, refer to the description of the JES Delays report in *z/OS RMF Report Analysis*.
5	5	UWDJS3MO	1	binary	JES3 modification code
Extended Data for HSM Section (See resource id in UWDUWRID):					
3	3	UWDHSMFU	1	binary	HSM function code For a list of HSM function codes, refer to the description of the HSM Delays report in *z/OS RMF Report Analysis*.
4	4	UWDHSMMO	1	binary	HSM modification code
Extended Data for ENQ Section (See resource id in UWDUWRID):					

Offsets		Name	Length	Format	Description
Dec	Hex				
3	3	UWDENTID	2	binary	ENQUEUE name table (ENTG3) index
5	5	UWDFLAGE	1	binary	ENQUEUE flags **Bit** **Meaning When Set** **0** OFF=Request is EXCLUSIVE ON=Request is SHARED **1** ON=Request from another system. (Fields UWDSYSNA/UWDJOBNA are valid) **2** Server name present **3-7** Reserved
6	6	UWDSASID	2	binary	Server address space analysis index. Valid if bit 2 of UWDFLAGE is set.
6	6	UWDSYSNA	8	EBCDIC	System name of requestor. Valid if bit 1 of UWDFLAGE is set.
14	E	UWDJOBNA	8	EBCDIC	Job name of requestor. Valid if bit 1 of UWDFLAGE is set.
Extended Data for MESSAGE section (See resource id in UWDUWRID):					
3	3	UWDOREID	4	EBCDIC	Reply number
Extended Data for MOUNT section (See resource id in UWDUWRID):					
3	3	UWDDEVIN	2	binary	DVTG3 table index
Extended Data for XCF section (See resource id in UWDUWRID):					
3	3	UWDXCDEV	4	EBCDIC	Device number of path on which the message is pending
7	7	UWDXCMAS	2	binary	ASID of member sending message
9	9	UWDXCHAS	2	binary	Name of ASID that initiated message out request

ERBXMHG3 - Moved samples header control block

Offsets		Name	Length	Format	Description
Dec	Hex				
0	0	XMHXMHG3	5	EBCDIC	Acronym 'XMHG3'
5	5	XMHRMFV	1	binary	Control block version X'03'
6	6	*	1	*	Reserved
7	7	XMHFLAG	1	binary	Flags **Bit** **Meaning When Set** **0** A data-set table was moved **1** No data-set table was moved **2** A DSNC3 table was moved **3** A DSNG3 table was moved **4-7** Reserved
8	8	XMHRETC	4	binary	Return codes **RC** **Meaning and Possible Environment** **0** Successful (XMEM and DS) **4** Time out of range (XMEM and DS) **8** Area too small (XMEM) **16** Severe error - dump call required (XMEM)
12	C	XMHLEN	4	binary	Total length of getmained sample area. If XMHRETC=8, total length needed to hold all data is returned here
12	C	XMHDSPTR	4	binary	Address of the sample area getmained by DS. Valid if XMHRETC=0 OFFSET TO FIRST SSH
16	10	XMHSSHFP	4	binary	Pointer to first SSHG3. This is an address within the requestor's address space.
20	14	XMHSSHLP	4	binary	Pointer to last SSHG3. This is an address within the requestor's address space.
24	18	XMHFRSTI	8	EBCDIC	Time of first SSH moved. Valid if XMHRETC = 0
32	20	XMHLSTTI	8	EBCDIC	Time of last SSH moved. Valid if XMHRETC = 0
40	28	XMHFRSTA	8	EBCDIC	Time of the first SSH available in the wrap around buffer
48	30	XMHLSTTA	8	EBCDIC	Time of the last SSH available in the wrap around buffer
56	38	XMHDSACI	2	binary	Index of the currently active data set within the DSNC3 data set names table
58	3A	*	2	*	Reserved
60	3C	XMHDSACL	8	EBCDIC	Time of the last SSH available on the active data set

End of Programming Interface information

Chapter 8. Monitor III data reporter tables

This topic provides the following information:

- It describes the data tables, and graphic parameter table used by the Monitor III data reporter
- It lists the ISPF record fields and table entries associated with creating, formatting, and displaying RMF reports

See Chapter 6, “Adding Monitor III user exits,” on page 157 for information on how to create user-defined reports.

Tabular report format table ERBFMTS3

Programming Interface information

The RMF format table defines the layout of RMF reports for panel display and hardcopy printing. It also ensures that each output function within RMF produces the same format.

This table contains one row for each report name and format. Each row contains information on how to edit heading and column data and contains an example for each variable name.

Variable Name	T	Variable Description	Example
FMTREPNA	K	Report name	DELAY
FMTFORMT	K	Report format identifier (not yet used)	ENGLISH
FMTRMODE	N	Report mode available (GRAPHIC/TABULAR/BOTH)	BOTH
FMTTPANL	N	Tabular report panel name	ERB3JDE
FMTTHLPP	N	Name of related help panel	ERB3JDE1
FMTLOGLN	N	Name of logical line number variable	JDEDTLLN
FMTSEQNR	N	Name of sequence number variable	JDEDTPSN
FMTCMDLN	N	Content of command line	COMMAND ===> &ZCMD ...
FMTHDR1	N	Content of header line 1 (text and variables intermixed)	... RMF DELAYS &HDRSID ..
FMTHDR2	N	Content of header line 2 (text and variables intermixed)	... Samples: &Z TIME: .
FMTSUBH1	N	Content of subheader line 1 (text and variables intermixed)	
FMTSUBH2	N	Content of subheader line 2 (text and variables intermixed)	
FMTSUBH3	N	Content of subheader line 3 (text and variables intermixed)	
FMTSUBH4	N	Content of subheader line 4 (text and variables intermixed)	

<table>
<tr><th>Variable Name</th><th>T</th><th>Variable Description</th><th>Example</th></tr>
<tr><td>FMTSUBH5</td><td>N</td><td>Content of subheader line 5 (text and variables intermixed)</td><td></td></tr>
<tr><td>FMTCOLH1</td><td>N</td><td>Text for column header line 1</td><td>WFL USG</td></tr>
<tr><td>FMTCOLH2</td><td>N</td><td>Text for column header line 2</td><td>NAME C DMN % % ..</td></tr>
<tr><td>FMTCOLH3</td><td>N</td><td>Text for column header line 3</td><td></td></tr>
<tr><td>FMTHVPRE</td><td>N</td><td>Prefix used in specifying variables in header lines</td><td>&</td></tr>
<tr><td>FMTHPLCH</td><td>N</td><td>Header line placeholder replacement variable names</td><td>HDRSAMPL HDRDATE HDRTIME</td></tr>
<tr><td>FMTSPLCH</td><td>N</td><td>Subheader line placeholder replacement variable names</td><td></td></tr>
<tr><td>FMTCPLCH</td><td>N</td><td>Command line placeholder replacement variable names</td><td>AMT</td></tr>
<tr><td>FMTMODL1</td><td>N</td><td>Definition of model line 1 (attribute characters followed by variable names or placeholder values(Z), variable names used must be elements of the report column data table)</td><td>|JDELDAN|Z|Z |Z</td></tr>
<tr><td>FMTMODL2</td><td>N</td><td>Definition of model line 2</td><td></td></tr>
<tr><td>FMTMODL3</td><td>N</td><td>Definition of model line 3</td><td></td></tr>
<tr><td>FMTMATTR</td><td>N</td><td>Attribute characters used in model lines</td><td>_|¢</td></tr>
<tr><td>FMTMPLCH</td><td>N</td><td>Model line placeholder replacement variable names (ZVARS)</td><td>JDETYPE JDELDMN JDELPGN</td></tr>
<tr><td>FMTHVMAX</td><td>N</td><td>Number of variables within header lines (maximum of 20)</td><td>6</td></tr>
<tr><td>FMTSVMAX</td><td>N</td><td>Number of variables within subheader lines (maximum of 30)</td><td>0</td></tr>
<tr><td>FMTMVMAX</td><td>N</td><td>Number of variables within model lines (maximum of 30)</td><td>16</td></tr>
<tr><td>FMTCVMAX</td><td>N</td><td>Number of variables within command line (maximum of 5)</td><td></td></tr>
<tr><td>FMTHVNnn</td><td>S</td><td>Variable name used in header lines</td><td>HDRSID</td></tr>
<tr><td>FMTHVRnn</td><td>S</td><td>Number of header line where variable is used</td><td>1</td></tr>
<tr><td>FMTHVPnn</td><td>S</td><td>Variable position within line</td><td>52</td></tr>
<tr><td>FMTHVLnn</td><td>S</td><td>Maximum variable length</td><td>15</td></tr>
<tr><td>FMTSVNxx</td><td>S</td><td>Variable name used in subheader lines</td><td></td></tr>
<tr><td>FMTSVRxx</td><td>S</td><td>Number of subheader line where variable is used</td><td></td></tr>
<tr><td>FMTSVPxx</td><td>S</td><td>Variable position within line</td><td></td></tr>
<tr><td>FMTSVLxx</td><td>S</td><td>Maximum variable length</td><td></td></tr>
</table>

Variable Name	T	Variable Description	Example
FMTMVNyy	S	Variable name used in model lines	JDELDAN
FMTMVRyy	S	Number of model line where variable is used	1
FMTMVPyy	S	Variable position within line	2
FMTMVLyy	S	Maximum variable length	8
FMTCVNzz	S	Variable name used in command line	ZCMD
FMTCVPzz	S	Variable position within line	14
FMTCVLzz	S	Maximum variable length	51

Note:

K - KEY type variable

N - NAMES type variable

S - EXTENSION type variable

nn = unique number for each variable used in the header lines

xx = unique number for each variable used in the subheader lines

yy = unique number for each variable used in the model lines

zz = unique number for each variable used in the command line

End of Programming Interface information

Header data table ERBHDRS3

Programming Interface information

The RMF header data table provides the variable heading information in one table row for each report.

Variable Name	T	Variable Description	Example
HDRREPNA	K	Report name	DELAY
ERBSID	N	System identifier	AQXA
ERBHCTXT	N	Hardcopy text constant	HARDCOPY
ERBSAMPL	N	Sample count	100
ERBDATE	N	Starting date	07/02/11
ERBTIME	N	Starting time	10.35.00
ERBRANGE	N	Time range value	100
ERBRMFVD	N	RMF version	RMF V1R13
ERBSPXID	N	Sysplex ID	RMFPLEX
ERBSNUM	N	Number of systems within sysplex	5
ERBSAMWL	N	Number of WLM samples	100

Variable Name	T	Variable Description	Example
	S	The variable data for subheader lines has to be kept in extension values of this table. Example for STORR report.	

Note:

K - KEY type variable

N - NAMES type variable

S - EXTENSION type variable

End of Programming Interface information

Monitor III data reporter tables

Programming Interface information

Each of the following report data tables indicates in column **Report** whether a value is part of the Monitor III report (Yes), is part of a pop-up window (Pop-Up), or is available through the Monitor III Utility (Util).

Column **T** indicates whether it is a KEY-type variable (K) or a NAMES-type variable (N).

CACHDET - Tabular report data table ERBCADT3

RMF builds ERBCADT3 when using CACHDET as a report type.

Name	T	Description of the Variable	Report
CADDTLLN	K	Logical line number	-
CADDTPSN	K	Sequence number	-
CADPVOLU	N	Volume	Yes
CADPDEVN	N	Device Number	Yes
CADPSSID	N	SSID	Yes
CADPIOP	N	I/O percentage	Yes
CADPIO	N	I/O rate	Yes
CADPHITP	N	Hit percentage	Yes
CADPREAD	N	Cache hit rate READ	Yes
CADPDFW	N	Cache hit rate DFW	Yes
CADPCFW	N	Cache hit rate CFW	Yes
CADPTOT	N	DASD I/O rate total	Yes
CADPSTAG	N	DASD I/O rate stage	Yes
CADPSEQ	N	Sequential rate	Yes
CADPASYN	N	Async rate	Yes
CADICACH	N	Cache state	Yes
CADIDFW	N	DFW state	Pop-Up
CADIPIN	N	Pinned state	Pop-Up
CADNRRA	N	Norm Read rate	Pop-Up
CADNRHI	N	Norm Read hit rate	Pop-Up
CADNRHIP	N	Norm Read hit percentage	Pop-Up
CADNWRA	N	Norm Write rate	Pop-Up
CADNWFA	N	Norm Write fast rate	Pop-Up
CADNWHI	N	Norm Write hit rate	Pop-Up
CADNWHIP	N	Norm Write hit percentage	Pop-Up
CADNREAP	N	Norm Read percentage	Pop-Up
CADNTRA	N	Norm Tracks rate	Pop-Up
CADSRRA	N	Seq Read rate	Pop-Up
CADSRHI	N	Seq Read hit rate	Pop-Up
CADSRHIP	N	Seq Read hit percentage	Pop-Up

Name	T	Description of the Variable	Report
CADSWRA	N	Seq Write rate	Pop-Up
CADSWFA	N	Seq Write fast rate	Pop-Up
CADSWHI	N	Seq Write hit rate	Pop-Up
CADSWHIP	N	Seq Write hit percentage	Pop-Up
CADSREAP	N	Seq Read percentage	Pop-Up
CADSTRA	N	Seq Tracks rate	Pop-Up
CADCRRA	N	CFW Read rate	Pop-Up
CADCRHI	N	CFW Read hit rate	Pop-Up
CADCRHIP	N	CFW Read hit percentage	Pop-Up
CADCWRA	N	CFW Write rate	Pop-Up
CADCWHI	N	CFW Write hit rate	Pop-Up
CADCWHIP	N	CFW Write hit percentage	Pop-Up
CADCREAP	N	CFW Read percentage	Pop-Up
CADTRRA	N	Total Read rate	Pop-Up
CADTRHI	N	Total Read hit rate	Pop-Up
CADTRHIP	N	Total Read hit percentage	Pop-Up
CADTWRA	N	Total Write rate	Pop-Up
CADTWFA	N	Total Write fast rate	Pop-Up
CADTWHI	N	Total Write hit rate	Pop-Up
CADTWHIP	N	Total Write hit percentage	Pop-Up
CADTREAP	N	Total Read percentage	Pop-Up
CADMDFWB	N	DFW bypass	Pop-Up
CADMNICL	N	Non-cache ICL	Pop-Up
CADMCWRI	N	CKD write	Pop-Up
CADMRCRM	N	Read miss	Pop-Up
CADMCFWB	N	CFW bypass	Pop-Up
CADMNBYP	N	Non-cache bypass	Pop-Up
CADMCHIT	N	CKD hits	Pop-Up
CADMRCWP	N	Write prom	Pop-Up
CADMDFWI	N	DFW inhibit	Pop-Up

CACHSUM - Tabular report data table ERBCAST3

RMF builds ERBCAST3 when using CACHSUM as a report type.

Name	T	Description of the Variable	Report
CASDTLLN	K	Logical line number	-
CASDTPSN	K	Sequence number	-
CASPSSID	N	SSID	Yes
CASPCUID	N	CUID	Yes
CASPTYPM	N	Type-Mod	Yes
CASPSIZE	N	Storage size	Yes

Name	T	Description of the Variable	Report
CASPIO	N	I/O rate	Yes
CASPHITP	N	Hit percentage	Yes
CASPHIT	N	Hit rate	Yes
CASPMTOT	N	Miss total rate	Yes
CASPMSTG	N	Miss stage rate	Yes
CASPREAP	N	Read percentage	Yes
CASPSEQ	N	Sequential rate	Yes
CASPASYN	N	Async rate	Yes
CASPOFF	N	Off rate	Yes
CASNRRA	N	Norm Read rate	Pop-Up
CASNRHI	N	Norm Read hit rate	Pop-Up
CASNRHIP	N	Norm Read hit percentage	Pop-Up
CASNWRA	N	Norm Write rate	Pop-Up
CASNWFA	N	Norm Write fast rate	Pop-Up
CASNWHI	N	Norm Write hit rate	Pop-Up
CASNWHIP	N	Norm Write hit percentage	Pop-Up
CASNREAP	N	Norm Read percentage	Pop-Up
CASNTRA	N	Norm Tracks rate	Pop-Up
CASSRRA	N	Seq Read rate	Pop-Up
CASSRHI	N	Seq Read hit rate	Pop-Up
CASSRHIP	N	Seq Read hit percentage	Pop-Up
CASSWRA	N	Seq Write rate	Pop-Up
CASSWFA	N	Seq Write fast rate	Pop-Up
CASSWHI	N	Seq Write hit rate	Pop-Up
CASSWHIP	N	Seq Write hit percentage	Pop-Up
CASSREAP	N	Seq Read percentage	Pop-Up
CASSTRA	N	Seq Tracks rate	Pop-Up
CASCRRA	N	CFW Read rate	Pop-Up
CASCRHI	N	CFW Read hit rate	Pop-Up
CASCRHIP	N	CFW Read hit percentage	Pop-Up
CASCWRA	N	CFW Write rate	Pop-Up
CASCWHI	N	CFW Write hit rate	Pop-Up
CASCWHIP	N	CFW Write hit percentage	Pop-Up
CASCREAP	N	CFW Read percentage	Pop-Up
CASTRRA	N	Total Read rate	Pop-Up
CASTRHI	N	Total Read hit rate	Pop-Up
CASTRHIP	N	Total Read hit percentage	Pop-Up
CASTWRA	N	Total Write rate	Pop-Up
CASTWFA	N	Total Write fast rate	Pop-Up
CASTWHI	N	Total Write hit rate	Pop-Up
CASTWHIP	N	Total Write hit percentage	Pop-Up

Name	T	Description of the Variable	Report
CASTREAP	N	Total Read percentage	Pop-Up
CASMCACH	N	Cache state	Pop-Up
CASMCCON	N	Cache configured	Pop-Up
CASMCAVL	N	Cache available	Pop-Up
CASMCOFF	N	Cache offline	Pop-Up
CASMCPIN	N	Cache pinned	Pop-Up
CASMNVS	N	NVS state	Pop-Up
CASMNCON	N	NVS configured	Pop-Up
CASMNPIN	N	NVS pinned	Pop-Up

CFACT - Tabular report data table ERBCFAT3

RMF builds ERBCFAT3 when using CFACT as a report type.

Name	T	Description of the Variable	Report
CFADTLLN	K	Logical line number	-
CFADTPSN	K	Sequence number	-
CFAPSTRU	N	Structure name	Yes
CFAPTYPE	N	Structure type	Yes
CFAPSTAT	N	Structure status	Yes
CFAPSTEX	N	Structure status extension	Util
CFAPSYS	N	System name	Yes
CFAPUTIP	N	CPU utilization %	Yes
CFAPSYNR	N	Sync rate	Yes
CFAPASS	N	Sync average service time	Yes
CFAPSYNC	N	Number of synchronous requests	Util
CFAPASYR	N	Async rate	Yes
CFAPAAS	N	Async average service time	Yes
CFAPASYC	N	Number of asynchronous requests	Util
CFAPACHG	N	Async changed %	Yes
CFAPADEL	N	Async delay %	Yes
CFAPQRT	N	Average queued request time	Util
CFAPCNVC	N	Converted request count	Util
CFAPDELC	N	Operation count delayed for dump serialization	Util
CFAPQUEC	N	Queued operation count	Util
CFAPMUSR	N	Maximum number of users	Util
CFAPTUSR	N	Total number of users	Util
CFAPPUSR	N	Number of problem users	Util
CFAPREBP	N	Rebuild percentage	Util
CFAINAM	N	Coupling facility name	Yes
CFAISTRU	N	Structure name	Pop-Up
CFAITYPE	N	Structure type	Pop-Up

Name	T	Description of the Variable	Report
CFAICNAM	N	Connection name	Pop-Up
CFAICJOB	N	Job name	Pop-Up
CFAICSTA	N	Status	Pop-Up
CFAICPRB	N	Problem status	Util
CFAICASI	N	ASID	Pop-Up
CFAICLVL	N	CF level	Pop-Up
CFAICREB	N	User managed rebuild allowed	Util
CFAICDRB	N	User managed rebuild with duplexing allowed	Util
CFAICALT	N	Altering allowed	Util
CFAICAUT	N	System managed processes allowed	Util
CFAICSUS	N	Suspension of work is tolerated	Util
CFAISTRS	N	Structure size	Pop-Up
CFAISTRP	N	Structure size %	Util
CFAISTUP	N	Structure utilized storage %	Util
CFAISTRC	N	Structure storage class	Util
CFAISTRM	N	Min structure size	Util
CFAISTRX	N	Max structure size	Util
CFAIDTS	N	Dump table size	Util
CFAILDES	N	Data element size (LIST/LOCK only)	Util
CFAILDLS	N	Data list entry size (LIST/LOCK only)	Util
CFAILEL	N	List entries total (LIST/LOCK only)	Pop-Up
CFAILEM	N	List entries current (LIST/LOCK only)	Pop-Up
CFAIMAE	N	Data elements total (LIST only)	Pop-Up
CFAICUE	N	Data elements current (LIST only)	Pop-Up
CFAILTL	N	Lock entries total (LIST/LOCK only)	Pop-Up
CFAILTM	N	Lock entries current (LIST/LOCK only)	Pop-Up
CFAIDES	N	Data element size (CACHE only)	Util
CFAIDEN	N	Directory entries total (CACHE only)	Pop-Up
CFAIDEC	N	Directory entries current (CACHE only)	Pop-Up
CFAIDEL	N	Data elements total (CACHE only)	Pop-Up
CFAIDAC	N	Data elements current (CACHE only)	Pop-Up
CFAICONT	N	Contention %	Pop-Up
CFAIFCON	N	False Contention % (LOCK only)	Pop-Up
CFAIREQR	N	Request rate (CACHE only)	Pop-Up
CFAIREAR	N	Read rate (CACHE only)	Pop-Up
CFAIWRIR	N	Write rate (CACHE only)	Pop-Up
CFAICAOR	N	Castout rate (CACHE only)	Pop-Up
CFAIXIR	N	XI rate (CACHE only)	Pop-Up
CFAIDER	N	Directory reclaims (CACHE only)	Pop-Up
CFAIFCCL	N	First castout class	Util
CFAILCCL	N	Last castout class	Util

Name	T	Description of the Variable	Report
CFAIPREF	N	Allocation preference list	Util
CFAIEXCL	N	Exclusion preference list	Util

CFOVER - Tabular report data table ERBCFOT3

RMF builds ERBCFOT3 when using CFOVER as a report type.

Name	T	Description of the Variable	Report
CFODTLLN	K	Logical line number	-
CFODTPSN	K	Sequence number	-
CFOPNAM	N	Coupling facility name	Yes
CFOPMOD	N	Model	Yes
CFOPVER	N	Version	Yes
CFOPLVL	N	CF level	Yes
CFOPDYND	N	CF dynamic dispatching	Yes
CFOPSTAT	N	Status of CF	Util
CFOPVOL	N	Volatility status	Util
CFOPUTIP	N	Processor utilization %	Yes
CFOPDEF	N	Processor defined	Yes
CFOPPDED	N	Number of dedicated processors	Util
CFOPPSHR	N	Number of shared processors	Yes
CFOPPWGT	N	Average weighting of shared processors	Yes
CFOPEFF	N	Processor effective	Yes
CFOPREQR	N	Request rate	Yes
CFOPTSD	N	Storage size	Yes
CFOPTSF	N	Storage available	Yes
CFOPUTIS	N	Utilized storage %	Util
CFOPTCS	N	Total control space	Util
CFOPFCS	N	Free control space	Util
CFOPDTS	N	Dump table control space	Util
CFOPDTUS	N	Dump table in use	Util
CFOPSYSC	N	Connected MVS system count	Util
CFOPSTCI	N	Structure count in policy	Util
CFOPSTCO	N	Structure count out policy	Util

CFSYS - Tabular report data table ERBCFST3

RMF builds ERBCFST3 when using CFSYS as a report type.

Name	T	Description of the Variable	Report
CFSDTLLN	K	Logical line number	-
CFSDTPSN	K	Sequence number	-
CFSPNAM	N	Coupling facility name	Yes

Name	T	Description of the Variable	Report
CFSPSYS	N	System name	Yes
CFSPSDEL	N	Subchannel delay %	Yes
CFSPSBSP	N	Subchannel busy %	Yes
CFSPPTHA	N	Paths available	Yes
CFSPPDEL	N	Paths delay %	Yes
CFSPSYNR	N	Sync rate	Yes
CFSPASS	N	Sync average service time	Yes
CFSPSYNC	N	Synchronous request count	Util
CFSPSOPD	N	Average synchronous operation delay	Util
CFSPSYNP	N	Synchronous request %	Util
CFSPASYR	N	Async rate	Yes
CFSPAAS	N	Async average service time	Yes
CFSPASYC	N	Asynchronous request count	Util
CFSPACHG	N	Async changed %	Yes
CFSPADEL	N	Async delay %	Yes
CFSPASYP	N	Asynchronous request %	Util
CFSPREQC	N	Total request %	Util
CFSPFOPT	N	Average failed operation time	Util
CFSPCNVC	N	Synchronous to asynchronous conversion rate	Util
CFSINAM	N	Coupling facility name	Pop-Up
CFSISCG	N	Subchannels generated	Pop-Up
CFSISCU	N	Subchannels in use	Pop-Up
CFSISCL	N	Subchannels max	Pop-Up
CFSIPATH	N	Paths IDs	Pop-Up
CFSIPTYP	N	'TYPES' variable string	Pop-up

CHANNEL - Tabular report data table ERBCHAT3

RMF builds ERBCHAT3 when using CHANNEL as a report type.

Name	T	Description of the Variable	Report
CHADTLLN	K	Logical line number	-
CHADTPSN	K	Sequence number	-
CHACPIVC	N	Channel path ID	Yes
CHACPNVC	N	Number of DCM-managed channels	Yes
CHACGVC	N	Channel type generation	Yes
CHACPTVC	N	Channel path type	Yes
CHACSIVC	N	Channel shared indication	Yes
CHACPUVC	N	Partition utilization percent	Yes
CHACTUVC	N	Total utilization percent	Yes
CHACTBVC	N	Bus utilization percent	Yes
CHACPRVC	N	Partition transfer rate (Read) in B/sec	Yes

Name	T	Description of the Variable	Report
CHACTRVC	N	Total transfer rate (Read) in B/sec	Yes
CHACPWVC	N	Partition transfer rate (Write) in B/sec	Yes
CHACTWVC	N	Total transfer rate (Write) in B/sec	Yes
CHACPMVC	N	Partition message sent rate	Util
CHACTMVC	N	Total message sent rate	Util
CHACPSVC	N	Partition message sent size	Util
CHACTSVC	N	Total message sent size	Util
CHACSFVC	N	Partition message sent fail rate	Util
CHACPFVC	N	Partition message receive fail rate	Util
CHACTFVC	N	Total message receive fail rate	Util
CHACFRTE	N	Rate of native FICON operations	Yes
CHACFACT	N	Average number of native FICON operations concurrently active	Yes
CHACXRTE	N	Rate of High Performance FICON (zHPF) operations	Yes
CHACXACT	N	Average number of zHPF operations concurrently active	Yes
CHACFDFR	N	Number of deferred native FICON operations per second	Util
CHACXDFR	N	Number of deferred zHPF operations per second	Util

CPC - Tabular report data table ERBCPCT3

RMF builds ERBCPCT3 when using CPC as a report type.

Name	T	Description of the Variable	Report
CPCDTLLN	K	Logical line number	-
CPCDTPSN	K	Sequence number	-
CPCPPNAM	N	Partition name	Yes
CPCPDMSU	N	Defined capacity limit	Yes
CPCPAMSU	N	Actual consumed MSUs	Yes
CPCPCAPI	N	Defined capping (yes/no/mix)	Yes
CPCPLPNO	N	Average number of logical processors	Yes
CPCPLEFU	N	Logical processor effective utilization %	Yes
CPCPLTOU	N	Logical processor total utilization %	Yes
CPCPPLMU	N	Physical LPAR utilization %	Yes
CPCPPEFU	N	Physical processor effective utilization %	Yes
CPCPPTOU	N	Physical processor total utilization %	Yes
CPCPIND	N	Type/partition indicator	No
CPCPLPND	N	Number of logical processors defined	Util
CPCPWGHT	N	Current weighting of shared CPU resources	Util
CPCPDEDP	N	Number of dedicated processors online	Util

Name	T	Description of the Variable	Report
CPCPLPSH	N	Percentage of the physical processor that a logical processor of the LPAR is entitled to use. If HiperDispatch is active, this is the percentage of logical processors with medium share.	Util
CPCPVCMH	N	If HiperDispatch is active, this is the number of logical processors with high share.	Util
CPCPVCMM	N	If HiperDispatch is active, this is the number of logical processors with medium share.	Util
CPCPVCML	N	If HiperDispatch is active, this is the number of logical processors with low share.	Util
CPCPOSNM	N	Operating system name	Util
CPCPLPCN	N	LPAR cluster name	Util
CPCPLCIW	N	Initial weight defined	Util
CPCPLCMW	N	Minimum weight defined	Util
CPCPLCXW	N	Maximum weight defined	Util
CPCPCGNM	N	Group capacity name	Util
CPCPCGLT	N	Group capacity limit	Util
CPCPCGEM	N	Group minimum entitlement	Util
CPCPCGEX	N	Group maximum entitlement	Util
CPCPCSMB	N	Central storage in MB	Util
CPCPUPID	N	User partition ID	Util

Fields in the CPC report header

Name	Description of the Variable	Report
CPCHPNAM	Name of partition that collected the data	Yes
CPCHMOD	Processor type	Yes
CPCHMDL	Processor model	Yes
CPCHCMSU	Effective CPC capacity (MSU/hour)	Yes
CPCHCCAI	Capacity adjustment indication	Util
CPCHCCCR	Capacity change reason	Util
CPCHWF	Weight % of Max	Yes
CPCHLMSU	4h MSU average	Yes
CPCHGNAM	Capacity group name	Yes
CPCHIMSU	Image capacity	Yes
CPCHCAP	WLM capping %	Yes
CPCHLMAX	4h MSU maximum	Yes
CPCHGLIM	Capacity group limit	Yes
CPCHGL4H	< 4h indicator for group	Yes
CPCHRMSU	Time until capping	Util
CPCHRGRP	Time until capacity group is subject to capping	Util
CPCHGAUN	Capacity group average unused service units	Util
CPCHCPU	CPC sequence number	Util

CPC data

Name	Description of the Variable	Report
CPCHCPNO	Number of physical CPs	Util
CPCHIFAN	Number of physical zAAPs	Util
CPCHICFN	Number of physical ICFs	Util
CPCHIFLN	Number of physical IFLs	Util
CPCHSUPN	Number of physical zIIPs	Util
CPCHPANO	Number of configured LPARs	Util
CPCHWAIT	Wait completion indicator	Util
CPCHPMSU	% capacity used	Util
CPCHDEDC	Number of dedicated CPs across CPC	Util
CPCHDEDA	Number of dedicated zAAPs across CPC	Util
CPCHDEDI	Number of dedicated zIIPs across CPC	Util
CPCHSHRC	Number of shared physical CPs across CPC	Util
CPCHSHRA	Number of shared physical zAAPs across CPC	Util
CPCHSHRI	Number of shared physical zIIPs across CPC	Util
CPCHVCPU	VARYCPU option (YES/NO)	Util
CPCHWMGT	WLM management (YES/NO)	Util

DELAY - Tabular report data table ERBJDET3

RMF builds ERBJDET3 when using DELAY as a report type.

Name	T	Description of the Variable	Report
JDEDTLLN	K	Logical line number	-
JDEDTPSN	K	Sequence number	-
JDELDAN	N	Jobname or summary	Yes
JDETYPE	N	Class (A, B, E, O, S, or T)	Util
JDETYPX	N	Class (A, B, E, O, S, or T) with possible extension O	Yes
JDELDMN	N	Domain number; no longer used	Yes
JDELPGN	N	Performance group number; no longer used	Yes
JDEPSVCL	N	Service class name	Yes
JDEGMIP	N	Indicator whether Storage Critical and/or CPU Critical (S, C, or SC)	Yes
JDELWFL	N	Work flow percentage	Yes
JDELUSG	N	Using percentage	Yes
JDELDEL	N	Delay percentage	Yes
JDELIDL	N	Idle percentage	Yes
JDELUKN	N	Unknown percentage	Yes
JDELPROC	N	Processor delay percentage	Yes
JDELDEV	N	Device delay percentage	Yes
JDELSTOR	N	Storage delay percentage	Yes
JDELSUBS	N	JES, HSM, and XCF delay percentage	Yes
JDELOPER	N	Operator delay percentage	Yes

Name	T	Description of the Variable	Report
JDELENQ	N	ENQ delay percentage	Yes
JDELJES	N	JES delay percentage	Util
JDELHSM	N	HSM delay percentage	Util
JDELXCF	N	XCF delay percentage	Util
JDELMNT	N	Operator mount delay percentage	Util
JDELMES	N	Operator message delay percentage	Util
JDELQUI	N	Operator quiesce delay percentage	Util
JDELREAS	N	Primary reason	Yes

DEV - Tabular report data table ERBDEVT3

RMF builds table ERBDEVT3 when using DEV as a report type.

Name	T	Description of the Variable	Report
DEVDTLLN	K	Logical line number	-
DEVDTPSN	K	Sequence number	-
DEVPJOB	N	Jobname	Yes
DEVPCLA	N	Class (A, B, O, S, or T)	Yes
DEVPDMN	N	Domain number; no longer used	Yes
DEVPPGN	N	Performance group number; no longer used	Yes
DEVPSVCL	N	Service class name	Yes
DEVPODEL	N	Overall delay percentage	Yes
DEVPOUSE	N	Overall using percentage	Yes
DEVPCON	N	Connect time	Yes
DEV1SDEL	N	Delay percentage causes by volser1	Yes
DEV1VOLU	N	Volume serial number volser1	Yes
DEV2SDEL	N	Delay percentage caused by volser2	Yes
DEV2VOLU	N	Volume serial number volser2	Yes
DEV3SDEL	N	Delay percentage cause by volser3	Yes
DEV3VOLU	N	Volume serial number volser3	Yes
DEV4SDEL	N	Delay percentage caused by volser4	Yes
DEV4VOLU	N	Volume serial number volser4	Yes

DEVR - Tabular report data table ERBDVRT3

RMF builds ERBDVRT3 when using DEVR as a report type.

Name	T	Description of the Variable	Report
DVRDTLLN	K	Logical line number	-
DVRDTPSN	K	Sequence number	-
DVRPVOLU	N	Volser	Yes
DVRPDEVN	N	Device number	Yes
DVRPIDEN	N	Device indication (model)	Yes

Name	T	Description of the Variable	Report
DVRPSTAT	N	Status	Yes
DVRPEXP	N	Number of exposures	Yes
DVRPACTV	N	Percentage of active time	Yes
DVRPCONN	N	Percentage of connect time	Yes
DVRPDISC	N	Percentage of disconnect time	Yes
DVRPPEND	N	Percentage of pending time	Util
DVRPDLYR	N	Pending delay reason header	Yes
DVRPDLYP	N	Pending delay reason percentage	Yes
DVRACTRT	N	Device activity rate	Yes
DVRRESPT	N	Response Time	Yes
DVRIOSQT	N	IOS queue time	Util
DVRPDVBT	N	Percentage of device busy delay time	Util
DVRPCUBT	N	Percentage of control unit busy delay time	Util
DVRPSPBT	N	Percentage of director port busy delay time	Util
DVRPJOBN	N	Jobname	Yes
DVRPCLA	N	Class (A, B, O, S, or T)	Yes
DVRPDMN	N	Domain number; no longer used	Yes
DVRPPGN	N	Performance group number; no longer used	Yes
DVRPSUSE	N	Percentage of using	Yes
DVRPSDEL	N	Percentage of delay	Yes
DVRPSVCL	N	Service class	Yes
DVRPKIND	N	Device type indicator	Util
DVRPLCUN	N	Logical control unit ID	Util

DI - Tabular report data table ERBDSIT3

RMF builds ERBDSIT3 when using DI as a report type.

Name	T	Description of the Variable	Report
DSIDTLLN	K	Logical line number	-
DSIDTPSN	K	Sequence number	-
DSI1SID	N	System identifier	Yes
DSI1DATE	N	Starting date	Yes
DSI1TIME	N	Starting time	Yes
DSI1DDNM	N	DD-name	Yes
DSI1DSNM	N	Data set name	Yes
DSI2DATE	N	Ending date	Yes
DSI2TIME	N	Ending time	Yes
DSI2MESS	N	Message field	Yes

DSND - Tabular report data table ERBDNDT3

RMF builds ERBDNDT3 when using DSND as a report type.

Name	T	Description of the Variable	Report
DNDDTLLN	K	Logical line number	-
DNDDTPSN	K	Sequence number	-
DNDPDSN	N	Data set name	Yes
DNDPVOLU	N	Volume serial	Yes
DNDPJOBN	N	Jobname	Yes
DNDPASID	N	ASID	Yes
DNDPDUSG	N	DUSG (Using %)	Yes
DNDPDDLY	N	DDLY (Delay %)	Yes

DSNJ - Tabular report data table ERBDNJT3

RMF builds ERBDNJT3 when using DSNJ as a report type.

Name	T	Description of the Variable	Report
DNJDTLLN	K	Logical line number	-
DNJDTPSN	K	Sequence number	-
DNJPASID	N	ASID	Yes
DNJPDSN	N	Data set name	Yes
DNJPVOLU	N	Volume	Yes
DNJPDEVN	N	Device number	Yes
DNJPDUSG	N	DUSG (Using %)	Yes
DNJPDDLY	N	DDLY (Delay %)	Yes

DSNV - Tabular report data table ERBDNVT3

RMF builds ERBDNVT3 when using DSNV as a report type.

Name	T	Description of the Variable	Report
DNVDTLLN	K	Logical line number	-
DNVDTPSN	K	Sequence number	-
DNVPDSN	N	Data set name	Yes
DNVPJOBN	N	Jobname	Yes
DNVPASID	N	ASID	Yes
DNVPDUSG	N	DUSG (Using %)	Yes
DNVPDDLY	N	DDLY (Delay %)	Yes

ENCLAVE - Tabular report data table ERBENCT3

RMF builds ERBENCT3 when using ENCLAVE as a report type.

Name	T	Description of the Variable	Report
ENCDTLLN	K	Logical line number	-

ENCLAVE data

Name	T	Description of the Variable	Report
ENCDTPSN	K	Sequence number	-
ENCENAME	N	Enclave name (generated)	Yes
ENCCLASS	N	Service class	Yes
ENCGOAL	N	Goal time	Yes
ENCGPERC	N	Goal percent	Yes
ENCPER	N	Period	Yes
ENCDENC	N	Dependent enclave indicator	Yes
ENCXENC	N	Multi-system enclave indicator	Yes
ENCTCPU	N	Total CPU time (seconds)	Yes
ENCTIFA	N	Total zAAP time (seconds)	Util
ENCTIFC	N	Total zAAP on CP time (seconds)	Util
ENCDCPU	N	Delta CPU time (seconds)	Pop-Up
ENCDIFA	N	Delta zAAP time (seconds)	Util
ENCDIFC	N	Delta zAAP on CP time (seconds)	Util
ENCDCPUP	N	Delta CPU percentage in Monitor III range	Yes
ENCDIFAP	N	Delta zAAP percentage	Util
ENCDIFCP	N	Delta zAAP on CP percentage	Util
ENCSAMP	N	Total execution samples	Pop-Up
ENCTUSG	N	% Total using samples	Yes
ENCTDLY	N	% Total delay samples	Yes
ENCIDLE	N	% Idle	Yes
ENCCUSG	N	% CPU using	Pop-Up
ENCIFAU	N	% zAAP using	Pop-Up
ENCIFCU	N	% zAAP on CP using	Util
ENCCDLY	N	% CPU delay	Pop-Up
ENCIUSG	N	% I/O using	Pop-Up
ENCIDLY	N	% I/O delay	Pop-Up
ENCIFAD	N	% zAAP delay	Pop-Up
ENCCCAP	N	% CPU capping	Pop-Up
ENCSTOR	N	% Storage delay	Pop-Up
ENCUNKN	N	% Unknown	Pop-Up
ENCQUED	N	% Queue delay	Pop-Up
ENCESTYP	N	Subsystem type	Pop-Up
ENCEOWNM	N	Owner name	Pop-Up
ENCEOSYS	N	Owner system	Pop-Up
ENCXTOKN	N	Export token	Pop-Up
ENCATTN	N	Number of attributes in table	Pop-Up
ENCATT01	N	Attribute 01	Pop-Up
ENCATT02	N	Attribute 02	Pop-Up
ENCATT03	N	Attribute 03	Pop-Up
ENCATT04	N	Attribute 04	Pop-Up

Name	T	Description of the Variable	Report
ENCATT05	N	Attribute 05	Pop-Up
ENCATT06	N	Attribute 06	Pop-Up
ENCATT07	N	Attribute 07	Pop-Up
ENCATT08	N	Attribute 08	Pop-Up
ENCATT09	N	Attribute 09	Pop-Up
ENCATT10	N	Attribute 10	Pop-Up
ENCATT11	N	Attribute 11	Pop-Up
ENCATT12	N	Attribute 12	Pop-Up
ENCATT13	N	Attribute 13	Pop-Up
ENCATT14	N	Attribute 14	Pop-Up
ENCATT15	N	Attribute 15	Pop-Up
ENCATT16	N	Attribute 16	Pop-Up
ENCTSUP	N	Total zIIP time (seconds)	Pop-Up
ENCTSUC	N	Total zIIP on CP time (seconds)	Util
ENCDSUP	N	Delta zIIP time (seconds)	Pop-Up
ENCDSUC	N	Delta zIIP on CP time (seconds)	Util
ENCDSUPP	N	Delta zIIP percentage	Util
ENCDSUCP	N	Delta zIIP on CP percentage	Util
ENCSUPU	N	% zIIP using	Pop-Up
ENCSUCU	N	% zIIP on CP using	Util
ENCSUPD	N	% zIIP delay	Pop-Up

ENQ - Tabular report data table ERBENQT3

RMF builds ERBENQT3 when using ENQ as a report type.

Name	T	Description of the Variable	Report
ENQDTLLN	K	Logical line number	-
ENQDTPSN	K	Sequence number	-
ENQPWJOB	N	Jobname of waiting job	Yes
ENQPODEL	N	Overall delay percentage	Yes
ENQPRDEL	N	Percentage of delay for the resource	Yes
ENQPWSTT	N	Status of waiting job	Yes
ENQPMAJS	N	Resource major name and scope or minor name	Yes
ENQPHDEL	N	Holding percentage for the holding job	Yes
ENQPHJOB	N	Jobname of holding job or system name for holding job	Yes
ENQPHSTT	N	Status for the holding job	Yes

ENQR - Tabular report data table ERBEQRT3

RMF builds ERBEQRT3 when using ENQR as a report type.

Name	T	Description of the Variable	Report
EQRDTLLN	K	Logical line number	-
EQRDTPSN	K	Sequence number	-
EQRPMAJS	N	Resource major name and scope or resource minor name	Yes
EQRPRDEP	N	Percentage of delay for the delayed job	Yes
EQRPWJOB	N	Jobname of delayed job	Yes
EQRPWSTT	N	Status of delayed job	Yes
EQRPHDEP	N	Holding percentage for the holding job	Yes
EQRPHJOB	N	Jobname of holding job or system name	Yes
EQRPHSTT	N	Status of holding job	Yes

HSM - Tabular report data table ERBHSMT3

RMF builds ERBHSMT3 when using HSM as a report type. The table variables are identical to the variables of the ERBJEST3 table; see the ERBJEST3 table for more information.

IOQUEUE - Tabular report data table ERBIOQT3

RMF builds ERBIOQT3 when using IOQUEUE as a report type.

Name	T	Description of the Variable	Report
IOQDTLLN	K	Logical line number	-
IOQDTPSN	K	Sequence number	-
IOQCPIVC	N	Channel path ID	Yes
IOQPATVC	N	Path attributes	Util
IOQDCMVC	N	DCM-managed channels	Yes
IOQPCUVC	N	Physical CU string	Yes
IOQMMNVC	N	Minimum number of DCM-managed channels used	Yes
IOQMMXVC	N	Maximum number of DCM-managed channels used	Yes
IOQMDFVC	N	Defined number of DCM-managed channels	Yes
IOQLCUVC	N	Logical control unit ID	Yes
IOQCRTVC	N	Contention rate	Yes
IOQDQLVC	N	Delay queue length	Yes
IOQCPTVC	N	Channel path ID taken	Yes
IOQSPBVC	N	Director port busy percent	Yes
IOQCUBVC	N	Control unit busy percent	Yes

JES - Tabular report data table ERBJEST3

RMF builds ERBJEST3 when using JES as a report type.

Name	T	Description of the Variable	Report
HJSDTLLN	K	Logical line number	-
HJSDTPSN	K	Sequence number	-
HJSPJOB	N	Jobname	Yes
HJSPODEL	N	Overall delay percentage	Yes
HJS1FDEL	N	Delay percentage	Yes
HJS1FCNR	N	Function code	Yes
HJS1EXPL	N	Explanation	Yes
HJS2FDEL	N	Delay percentage	Util
HJS2FCNR	N	Function code	Util
HJS2EXPL	N	Explanation	Util

JOB - Tabular report data table ERBJDJT3

RMF builds ERBJDJT3 when using JOB as a report type.

Name	T	Description of the Variable	Report
JDJDTLLN	K	Logical line number	-
JDJDTPSN	K	Sequence number	-
JDJLDAN	N	Jobname or summary	Yes
JDJLASID	N	Address space identification	Yes
JDJCLASS	N	Class (A, B, E, O, S, or T)	Util
JDJCLASX	N	Class (A, B, E, O, S, or T) with possible extension O	Yes
JDJLDMN	N	Domain number; no longer used	Yes
JDJLPGN	N	Performance group number; no longer used	Yes
JDJPSVCL	N	Service class name	Yes
JDJLWFL	N	Work flow percentage	Yes
JDJLUSP	N	Processor using percentage	Yes
JDJLUSD	N	Device using percentage	Yes
JDJLUSG	N	Using percentage	Util
JDJLDEL	N	Delay percentage	Yes
JDJLIDL	N	Idle percentage	Yes
JDJLUKN	N	Unknown percentage	Yes
JDJLPROC	N	Processor delay percentage	Yes
JDJLDEV	N	Device delay percentage	Yes
JDJLSTOR	N	Storage delay percentage	Yes
JDJLSUBS	N	SUBS delay percentage	Yes
JDJLOPER	N	Operator delay percentage	Yes
JDJLENQ	N	ENQ delay percentage	Yes
JDJLJES	N	JES delay percentage	Util

Name	T	Description of the Variable	Report
JDJLHSM	N	HSM delay percentage	Util
JDJLXCF	N	XCF delay percentage	Util
JDJLMNT	N	Operator mount delay percentage	Util
JDJLMES	N	Operator message delay percentage	Util
JDJLQUI	N	Operator quiesce delay percentage	Util
JDJLREAS	N	Primary reason	Yes

LOCKSP - Tabular report data table ERBLSPT3

RMF builds ERBLSPT3 when using LOCKSP as a report type.

Name	T	Description of the Variable	Report
LSPDTLLN	K	Logical line number	-
LSPDTPSN	K	Sequence number	-
LSPPRES	N	Resource name	Yes
LSPPJT	N	Spinner jobname / spin lock type	Yes
LSPPAC	N	Spinner address space ID / CPU ID	Yes
LSPPRAD	N	Spinner request address	Yes
LSPPHELD	N	Spin lock held percentage	Yes
LSPPSPIN	N	Spin percentage	Yes

LOCKSU - Tabular report data table ERBLSUT3

RMF builds ERBLSUT3 when using LOCKSU as a report type.

Name	T	Description of the Variable	Report
LSUDTLLN	K	Logical line number	-
LSUDTPSN	K	Sequence number	-
LSUPRES	N	Resource name	Yes
LSUPTYPE	N	Lock Type	Yes
LSUPJOB	N	Holder job name	Yes
LSUPASI	N	Holder address space ID	Yes
LSUPRAD	N	Request address	Yes
LSUPHELD	N	Holding percentage	Yes
LSUPINTR	N	Holding while interrupted percentage	Yes
LSUPDISP	N	Holding while dispatched percentage	Yes
LSUPSUSP	N	Holding while suspended percentage	Yes

OPD - Tabular report data table ERBOPDT3

RMF builds ERBOPDT3 when using OPD as a report type.

Name	T	Description of the Variable	Report
OPDDTLLN	K	Logical line number	-
OPDDTPSN	K	Sequence number	-

Name	T	Description of the Variable	Report
OPDPJOBN	N	Jobname	Yes
OPDPUSEN	N	User name	Yes
OPDPASID	N	ASID	Yes
OPDPASIX	N	Hexadecimal ASID	Yes
OPDPPRID	N	Process ID	Yes
OPDPPPID	N	Parent's process ID	Yes
OPDPLATW	N	Waiting for process latch	Yes
OPDPSTAT	N	Process state	Yes
OPDPAPPL	N	Percentage of TCB and SRB time	Yes
OPDPTOT	N	Total computing time since process has been started	Yes
OPDPSERV	N	Server type	Yes
OPDIPRID	N	Process ID	Pop-up
OPDIPPID	N	Parent's process ID	Pop-up
OPDIJOBN	N	Jobname	Pop-up
OPDIUSEN	N	User name	Pop-up
OPDIASID	N	ASID	Pop-up
OPDIASIX	N	Hexadecimal ASID	Pop-up
OPDITIMD	N	Start time/date	Pop-up
OPDIAPPL	N	Percentage of TCB and SRB time	Pop-up
OPDITOT	N	Total computing time since process has been started	Pop-up
OPDILPID	N	Latch process ID the process is waiting for (0 = not waiting)	Pop-up
OPDICMD	N	Command buffer	Pop-up
OPDISERN	N	Server name	Pop-up
OPDISERV	N	Server type	Pop-up
OPDIACTF	N	Number of active files	Pop-up
OPDIMAXF	N	Maximum files	Pop-up
OPDISTAT	N	Process state	Pop-up
OPDISTA1	N	1. possible state	Pop-up
OPDISTA2	N	2. possible state	Pop-up
OPDISTA3	N	3. possible state	Pop-up

PROC - Tabular report data table ERBPRCT3

RMF builds ERBPRCT3 when using PROC as a report type.

Name	T	Description of the Variable	Report
PRCDTLLN	K	Logical line number	-
PRCDTPSN	K	Sequence number	-
PRCPJOB	N	Jobname	Yes
PRCPASI	N	Address space ID of the job (decimal format)	Util

Name	T	Description of the Variable	Report
PRCPCLA	N	Class (A, B, E, O, S, or T)	Util
PRCPCLAX	N	Class (A, B, E, O, S, or T) with possible extension O	Yes
PRCPDMN	N	Domain number; no longer used	Yes
PRCPPGN	N	Performance group number; no longer used	Yes
PRCPSVCL	N	Service class name	Yes
PRCPODEL	N	Overall delay percentage for this address space.	Util
PRCPOUSE	N	Overall using percentage for this address space.	Util
PRCPTYPE	N	Processor type	Yes
PRCPTST	N	Overall application percentage for this address space.	Util
PRCPCAP	N	Capping delay percentage	Util
PRCPETST	N	Overall application percentage including EAppl percentage	Util
PRCPAPPL	N	Overall application percentage on behalf of this address space and processor type	Util
PRCPEAPP	N	Overall application percentage including EAppl percentage on behalf of this address space and processor type	Yes
PRCPTWFL	N	Overall workflow percentage of this address space and processor type	Util
PRCPTDEL	N	Overall delay percentage for this address space and processor type	Yes
PRCPTUSE	N	Overall using percentage for this address space and processor type	Yes
PRCPAACP	N	% zAAP on CP using	Util
PRCPIICP	N	% zIIP on CP using	Util
PRC1SDEL	N	Delay percentage caused by jobname1	Yes
PRC1JOBN	N	Jobname1	Yes
PRC2SDEL	N	Delay percentage caused by jobname2	Yes
PRC2JOBN	N	Jobname2	Yes
PRC3SDEL	N	Delay percentage caused by jobname3	Yes
PRC3JOBN	N	Jobname3	Yes
PRCTCPUT	N	Total CPU time (milliseconds)	Util

PROCU - Tabular report data table ERBPRUT3

RMF builds ERBPRUT3 when using PROCU as a report type.

Name	T	Description of the Variable	Report
PRUDTLLN	K	Logical line number	-
PRUDTPSN	K	Sequence number	-
PRUPJOB	N	Jobname	Yes
PRUPASI	N	Address space ID of the job (decimal format)	Yes
PRUPCLA	N	Class (A, B, E, O, S, or T)	Util

Name	T	Description of the Variable	Report
PRUPCLAX	N	Class (A, B, E, O, S, or T) with possible extension O	Yes
PRUPSVCL	N	Service class name	Yes
PRUPCLP	N	Service class period	Util
PRUPCPT	N	Time on CP %	Yes
PRUPAACT	N	zAAP time on CP %	Yes
PRUPIICT	N	zIIP time on CP %	Yes
PRUPCPE	N	CP EAppl %	Yes
PRUPAAPE	N	zAAP EAppl %	Yes
PRUPIIPE	N	zIIP EAppl %	Yes
PRUPTOTC	N	Percentage of total accumulated CPU time as sum of TCB time, global and local SRB time and preemptable or client SRB time, consumed on behalf of this address space.	Util
PRUPTOTE	N	Percentage of total accumulated CPU time as sum of TCB time, global and local SRB time, preemptable or client SRB time, and enclave time consumed within this address space.	Util
PRUPTCB	N	Percentage of TCB time consumed in this address space.	Util
PRUPSRB	N	Percentage of SRB time consumed in this address space by local or global SRBs.	Util
PRUPPCS	N	Percentage of preemptable or client SRB time consumed on behalf of this address space.	Util
PRUPEPS	N	Percentage of preemptable or client SRB and enclave CPU time consumed within this address space.	Util
PRUTCPUT	N	Total CPU time (milliseconds)	Util

RLSDS - Tabular report data table ERBVRDT3

RMF builds ERBVRDT3 when using RLSDS as a report type.

Name	T	Description of the Variable	Report
VRDDTLLN	K	Logical line number	-
VRDDTPSN	K	Sequence number	-
VRDPNAM	N	VSAM sphere name, Data set name, MVS system name, Access type, Response time, Read rate, Read BMF hit percentage, Read CF hit percentage	Yes
VRDPRDAS	N	Read DASD hit percentage	Yes
VRDPBMFV	N	BMF valid percentage	Yes
VRDPBMFF	N	BMF false invalid percentage	Yes
VRDPWRTE	N	Write rate	Yes
VRDPCALO	N	Castout lock percentage	Util
VRDPREDA	N	Redo activity percentage	Util
VRDPRRED	N	Recursive redo percentage	Util

Name	T	Description of the Variable	Report
VRDPIND	N	Report indication	Util

RLSLRU - Tabular report data table ERBVRLT3

RMF builds ERBVRLT3 when using RLSLRU as a report type.

Name	T	Description of the Variable	Report
VRLDTLLN	K	Logical line number	-
VRLDTPSN	K	Sequence number	-
VRLPSYS	N	MVS system name	Yes
VRLPACT	N	Average CPU time	Yes
VRLPBSG	N	Buffer size goal	Yes
VRLPBSH	N	Buffer size high	Yes
VRLPBSO	N	Buffer percentage accelerated	Yes
VRLPBSS	N	Buffer percentage reclaiming	Yes
VRLPABMF	N	Average BMF hit percentage	Yes
VRLPACAC	N	Average Cache hit percentage	Yes
VRLPADAS	N	Average DASD hit percentage	Yes
VRLPCALO	N	Castout lock percentage	Util
VRLPREDA	N	Redo activity percentage	Util
VRLPRRED	N	Recursive redo percentage	Util
VRLISYS	N	MVS system name	Pop-Up
VRLIFPL	N	Fixed pages low	Pop-Up
VRLIFPH	N	Fixed pages high	Pop-Up
VRLIFPA	N	Fixed pages average	Pop-Up
VRLIFIXS	N	Fixed storage	Pop-Up
VRLIRSP	N	Real storage %	Pop-Up
VRLILSn	N	Buffer count by pool of size nK, low value, below 2 GB, where n is 2, 4, ..., 32,	Pop-Up
VRLIHSn	N	Buffer count by pool of size nK, high value, below 2 GB, where n is 2, 4, ..., 32,	Pop-Up
VRLICSn	N	Buffer count by pool of size nK, average value, below 2 GB, where n is 2, 4, ..., 32	Pop-Up
VRLAFPL	N	Fixed pages low above 2 GB	Pop-Up
VRLAFPH	N	Fixed pages high above 2 GB	Pop-Up
VRLAFPA	N	Fixed pages average above 2 GB	Pop-Up
VRLAFIXS	N	Fixed storage above 2 GB	Pop-Up
VRLARSP	N	Real storage % above 2 GB	Pop-Up
VRLALSn	N	Buffer count by pool of size nK, low value, above 2 GB, where n is 2, 4, ..., 32,	Pop-Up
VRLAHSn	N	Buffer count by pool of size nK, high value, above 2 GB, where n is 2, 4, ..., 32,	Pop-Up
VRLACSn	N	Buffer count by pool of size nK, average value, above 2 GB, where n is 2, 4, ..., 32,	Pop-Up

RLSSC - Tabular report data table ERBVRST3

RMF builds ERBVRST3 when using RLSSC as a report type.

Name	T	Description of the Variable	Report
VRSDTLLN	K	Logical line number	-
VRSDTPSN	K	Sequence number	-
VRSPNAM	N	Storage class name, MVS system name, CF structure name, Access type	Yes
VRSPRTIM	N	Average response time	Yes
VRSPRRTE	N	Read rate	Yes
VRSPRBMF	N	Read BMF hit percentage	Yes
VRSPRCF	N	Read CF hit percentage	Yes
VRSPRDAS	N	Read DASD hit percentage	Yes
VRSPBMFV	N	BMF valid percentage	Yes
VRSPBMFF	N	BMF false invalid percentage	Yes
VRSPWRTE	N	Write rate	Yes
VRSPIND	N	Report indication	Util

SPACED - Tabular report data table ERBSPDT3

RMF builds ERBSPDT3 when using SPACED as a report type.

Name	T	Description of the Variable	Report
SPDDTLLN	K	Logical line number	-
SPDDTPSN	K	Sequence number	-
SPDPVOL	N	Volume	Yes
SPDPTSP	N	Total capacity in MB	Yes
SPDPFSP	N	Free space in MB	Yes
SPDPFSR	N	Free space percentage	Yes
SPDPLBK	N	Largest block in MB	Yes
SPDPSGN	N	Storage group name	Yes

SPACEG - Tabular report data table ERBSPGT3

RMF builds ERBSPGT3 when using SPACEG as a report type.

Name	T	Description of the Variable	Report
SPGDTLLN	K	Logical line number	-
SPGDTPSN	K	Sequence number	-
SPGPSGN	N	Storage Group	Yes
SPGPTSP	N	Total capacity in MB	Yes
SPGPFSP	N	Free space in MB	Yes
SPGPFSR	N	Free space percentage	Yes
SPGPNVO	N	Number of Volumes	Yes

STOR - Tabular report data table ERBSTRT3

RMF builds ERBSTRT3 when using STOR as a report type.

Name	T	Description of the Variable	Report
STRDTLLN	K	Logical line number	-
STRDTPSN	K	Sequence number	-
STRPJOB	N	Jobname	Yes
STRPCLA	N	Class (A, B, O, S, or T)	Yes
STRPDMN	N	Domain number; no longer used	Yes
STRPPGN	N	Performance group number; no longer used	Yes
STRPSVCL	N	Service class name	Yes
STRPODEL	N	Overall delay percentage	Yes
STR1SDEL	N	Delay percentage COMM	Yes
STR2SDEL	N	Delay percentage local	Yes
STR3SDEL	N	Delay percentage VIO	Util
STR4SDEL	N	Delay percentage SWAP	Yes
STR5SDEL	N	Delay percentage OUTR	Yes
STR6SDEL	N	Cross memory delay %	Util
STR7SDEL	N	Hiperspace delay %	Util
STR8SDEL	N	Other delays % (including VIO, XMEM and HIPR)	Yes
STRPACTV	N	Average ACTV frames	Util
STRPFIXD	N	Average fixed frames total	Util
STRPIDLE	N	Average IDLE frames	Util
STRPWSET	N	Average working set frames	Yes
STRPWSEX	N	Average ES working set frames	Yes

STORC - Tabular report data table ERBCSUT3

RMF builds ERBCSUT3 when using STORC as a report type.

Name	T	Description of the Variable	Report
CSUDTLLN	K	Logical line number	-
CSUDTPSN	K	Sequence number	-
CSXNAME	N	Jobname	Yes
CSXACT	N	Active column	Yes
CSXCLA	N	Class (A, B, O, S, or T)	Yes
CSXDMN	N	Domain number; no longer used	Yes
CSXPGN	N	Performance group number; no longer used	Yes
CSXCSN	N	Service class name	Yes
CSXASID	N	Address space identifier	Yes
CSXTIME	N	Elapsed time	Yes
CSXPCSA	N	Percentage of CSA	Yes
CSXPECS	N	Percentage of ECSA	Yes
CSXPSQA	N	Percentage of SQA	Yes

Name	T	Description of the Variable	Report
CSXPESQ	N	Percentage of ESQA	Yes
CSXACSA	N	Amount of CSA	Yes
CSXAECS	N	Amount of ECSA	Yes
CSXASQA	N	Amount of SQA	Yes
CSXAESQ	N	Amount of ESQA	Yes
CSXJESID	N	JES identifier	Util
CSXTDATE	N	Termination date	Util
CSXTTIME	N	Termination time	Util

STORCR - Tabular report data table ERBCRST3

RMF builds ERBCRST3 when using STORCR as a report type.

Name	T	Description of the Variable	Report
CSUDTLLN	K	Logical line number	-
CSUDTPSN	K	Sequence number	-
CSXNAME	N	Jobname	Yes
CSXJESID	N	JES identifier	Yes
CSXTDATE	N	Termination date	Yes
CSXTTIME	N	Termination time	Yes
CSXACSA	N	Amount of CSA	Yes
CXSAECS	N	Amount of ECSA	Yes
CSXASQA	N	Amount of SQA	Yes
CSXAESQ	N	Amount of ESQA	Yes

STORF - Tabular report data table ERBSTFT3

RMF builds ERBSTFT3 when using STORF as a report type.

Name	T	Description of the Variable	Report
STFDTLLN	K	Logical line number	-
STFDTPSN	K	Sequence number	-
STFPJOB	N	Jobname	Yes
STFPCLA	N	Class (A, B, O, S, or T)	Yes
STFPDMN	N	Domain number; no longer used	Yes
STFPPGN	N	Performance group number; no longer used	Yes
STFPSVCL	N	Service class name	Yes
STFPTOTL	N	Frame occupancy TOTAL	Yes
STFPACTV	N	Frame occupancy ACTV	Yes
STFPIDLE	N	Frame occupancy IDLE	Yes
STFPWSET	N	Active frames WSET	Yes
STFPFIXD	N	Active frames FIXED	Yes
STFPDIV	N	Active frames DIV	Yes

Name	T	Description of the Variable	Report
STFPAUXS	N	Auxiliary storage slots	Yes
STFPPGIN	N	Page-in Rate	Yes
STFPEXIN	N	Page-in rate from expanded storage	Util
STFPSPPI	N	Shared pages page-in rate from auxiliary storage	Util
STFPTOTS	N	Total number of shared page views	Util
STFPSVIN	N	Total number of valid shared pages	Util
STFPSPVL	N	Shared pages validation rate	Util
STFPLMO	N	Number of memory objects allocated	Util
STFPLPR	N	Number of 1 MB frames backed in real storage	Yes

STORM - Tabular report data table ERBSTMT3

RMF builds ERBSTMT3 when using STORM as a report type.

Name	T	Description of the Variable	Report
STMDTLLN	K	Logical line number	-
STMDTPSN	K	Sequence Number	-
STMPJOB	N	Jobname	Yes
STMPASI	N	Address space identifier	Yes
STMPCLA	N	Class (A, B, O, S, or T)	Yes
STMPSVCL	N	Service class name	Yes
STMPCLP	N	Service class period	Util
STMPTMO	N	Average number of memory objects allocated with this address space as the owner	Yes
STMPCMO	N	Average number of 64-bit common memory objects allocated with this address space as the owner	Yes
STMPSMO	N	Average number of shared memory objects allocated with this address space as the owner	Yes
STMPPMO	N	Average number of private memory objects allocated with this address space as the owner	Util
STMPLMO	N	Average number of large memory objects allocated with this address space as the owner	Yes
STMPLFR	N	Average number of 1 MB frames backed in real storage owned by this address space	Yes
STMPVTB	N	Average amount of storage allocated from large virtual memory in memory objects with this address space as the owner (in MB)	Yes
STMPCMB	N	Average amount of 64-bit common storage allocated with this address space as the owner (in MB)	Yes
STMPVSB	N	Average amount of shared storage allocated from large virtual memory in memory objects with this address space as the owner (in MB)	Yes
STMPPMB	N	Average amount of private storage allocated with this address space as the owner (in MB)	Util

Name	T	Description of the Variable	Report
STMPHCB	N	High water mark for the amount of 64-bit common storage allocated with this address space as the owner	Util
STMPLMB	N	Address space memory limit in MB	Util

STORR - Tabular report data table ERBSRRT3

RMF builds ERBSRRT3 when using STORR as a report type.

Name	T	Description of the Variable	Report
SRRDTLLN	K	Logical line number	-
SRRDTPSN	K	Sequence number	-
SRRVOLVC	N	Volume serial number	Yes
SRRDEVTY	N	Device type	Yes
SRRCUTY	N	Control unit type	Yes
SRREXPCT	N	Number of exposures	Yes
SRRUSVC	N	Percentage of using	Util
SRRA1VC	N	Percentage of active	Yes
SRRA2VC	N	Percentage of connect	Yes
SRRA3VC	N	Percentage of disconnect	Yes
SRRA4VC	N	Percentage of pending	Yes
SRRA5VC	N	Percentage of DLY-DB	Util
SRRA6VC	N	Percentage of DLY-CUB	Util
SRRA7VC	N	Percentage of DLY-SPB	Util
SRRSPTVC	N	Space type	Yes
SRRAUTOT	N	Average active users: TOTAL	Yes
SRRAULOC	N	Average active users: LOCAL	Yes
SRRAUSWP	N	Average active users: SWAP	Yes
SRRAUCOM	N	Average active users: COMM	Yes
SRRPDLYR	N	Delay type header	Yes
SRRPDLYP	N	Delay reason percentage	Util

STORS - Tabular report data table ERBSRST3

RMF builds ERBSRST3 when using STORS as a report type.

Name	T	Description of the Variable	Report
SRSDTLLN	K	Logical line number	-
SRSDTPSN	K	Sequence number	-
SRSPDMPG	N	WLM group name for graphic report	Yes
SRSPDMN	N	Domain number; no longer used	Yes
SRSPPGN	N	Performance group number; no longer used	Yes
SRSPGNAM	N	Name of WLM group	Yes
SRSPGTYP	N	Type of WLM group	Yes

Name	T	Description of the Variable	Report
SRSPTOTU	N	Total number of users	Yes
SRSPACTU	N	Number of active users	Yes
SRS1SDEL	N	Average number delayed for ANY	Yes
SRS2SDEL	N	Average number delayed for COMM	Yes
SRS3SDEL	N	Average number delayed for LOCL	Yes
SRS4SDEL	N	Average number delayed for VIO	Util
SRS5SDEL	N	Average number delayed for SWAP	Yes
SRS6SDEL	N	Average number delayed for OUTR	Yes
SRS7SDEL	N	Average number delayed for cross memory	Util
SRS8SDEL	N	Average number delayed for hiperspace	Util
SRS9SDEL	N	Average number delayed for other reasons, including VIO, XMEM and HIPR	Yes
SRSPACTV	N	Average ACTV frames	Yes
SRSPFIXD	N	Average FIXED frames	Yes
SRSPIDLE	N	Average IDLE frames	Yes
SRSPPGIN	N	Page-in rate	Yes

SYSENQ - Tabular report data table ERBEQST3

RMF builds ERBEQST3 when using SYSENQ as a report type.

Name	T	Description of the Variable	Report
EQSDTLLN	K	Logical line number	-
EQSDTPSN	K	Sequence number	-
EQSPMAJN	N	Resource major name or resource minor name	Yes
EQSPWDEP	N	Percentage of delay for the delayed job	Yes
EQSPWJOB	N	Jobname of delayed job	Yes
EQSPWSNM	N	MVS system name of delayed job	Yes
EQSPWSTT	N	Status of delayed job	Yes
EQSPHUSP	N	Holding percentage for the holding job	Yes
EQSPHJOB	N	Jobname of holding job	Yes
EQSPHSNM	N	MVS system name of holding job	Yes
EQSPHSTT	N	Status of holding job	Yes

SYSINFO - Tabular report data table ERBSYST3

RMF builds ERBSYST3 when using SYSINFO as a report type.

Name	T	Description of the Variable	Report
SYSDTLLN	K	Logical line number	-
SYSDTPSN	K	Sequence number	-
SYSNAMVC	N	WLM group name	Yes
SYSTYPVC	N	Type of WLM group	Yes
SYSWFLVC	N	Workflow percentage	Yes

Name	T	Description of the Variable	Report
SYSTUSVC	N	Average number of total users	Yes
SYSAUSVC	N	Average number of active users	Yes
SYSTRSVC	N	Transactions / sec	Yes
SYSAFCVC	N	Active frames percentage	Util
SYSVECVC	N	Vector utilization	Util
SYSAUPVC	N	Average number using PROC	Yes
SYSAUDVC	N	Average number using DEV	Yes
SYSADPVC	N	Average number delayed for PROC	Yes
SYSADDVC	N	Average number delayed for DEV	Yes
SYSADSVC	N	Average number delayed for STOR	Yes
SYSADUVC	N	Average number delayed for SUBS	Yes
SYSADOVC	N	Average number delayed for OPER	Yes
SYSADEVC	N	Average number delayed for ENQ	Yes
SYSADJVC	N	Average number delayed for JES	Util
SYSADHVC	N	Average number delayed for HSM	Util
SYSADXVC	N	Average number delayed for XCF	Util
SYSADNVC	N	Average number delayed for Mount	Util
SYSADMVC	N	Average number delayed for Message	Util
SYSCPUVC	N	Total CPU time (TCB+SRB) percentage in the Monitor III range	Util
SYSSRBVC	N	Total SRB time percentage in the Monitor III range	Util
SYSTCBVC	N	Total TCB time percentage in the Monitor III range	Util
SYSIFAVC	N	Total zAAP time percentage in the Monitor III range	Util
SYSCPVC	N	Total standard CP time percentage in the Monitor III range	Util
SYSIFCVC	N	Total zAAP on standard CP time percentage in the Monitor III range	Util
SYSRSPVC	N	Average response time per transaction	Yes
SYSVELVC	N	Execution velocity	Util
SYSUGMVC	N	% using	Util
SYSUGPVC	N	% using processor	Util
SYSUGDVC	N	% using device	Util
SYSWGDVC	N	% device workflow	Util
SYSWGPVC	N	% processor workflow	Util
SYSDGMVC	N	% delay	Util
SYSUJMVC	N	Average number users using	Util
SYSDJMVC	N	Average number users delayed	Util
SYSDGEVC	N	% delay for ENQ	Util
SYSDGHVC	N	% delay for HSM	Util
SYSDGDVC	N	% delay for DEV	Util
SYSDGJVC	N	% delay for JES	Util

Name	T	Description of the Variable	Report
SYSDGOVC	N	% delay for OPER	Util
SYSDGPVC	N	% delay for PROC	Util
SYSDGSVC	N	% delay for STOR	Util
SYSDGUVC	N	% delay for SUBS	Util
SYSDGXVC	N	% delay for XCF	Util
SYSSUPVC	N	Total zIIP time percentage in the Monitor III range	Util
SYSSUCVC	N	Total zIIP on standard CP time percentage in the Monitor III range	Util
SYSPDPVC	N	CPU time in seconds, that transactions of a class or group were running at a promoted dispatching priority during the report interval.	Util
SYSTODVC	N	% of total delay samples	Util
SYSCPDVC	N	% of CP delay samples	Util
SYSAPDVC	N	% of zAAP delay samples	Util
SYSIPDVC	N	% of zIIP delay samples	Util
SYSRGCVC	N	CPU capping because resource group maximum being enforced	Util

Fields in the SYSINFO report header

Name	Description of the Variable	Report
SYSPARVC	Partition name	Yes
SYSMODVC	Processor family	Yes
SYSMDLVC	Processor model	Yes
SYSTSVVC	Appl% CP	Yes
SYSPOLVC	Policy name	Yes
SYSPRVVC	Average number of online CPs	Yes
SYSCUVVC	Average CPU Util% of CPs	Yes
SYSTSEVC	EAppl% CP	Yes
SYSPADVC	Policy date	Yes
SYSPRIVC	Average number of online zAAPs	Yes
SYSLCPVC	Average MVS Util% of CPs	Yes
SYSAPIVC	Appl% zAAP	Yes
SYSPATVC	Policy time	Yes
SYSPRTVC	Average number of online zIIPs	Yes
SYSAPTVC	Appl% zIIP	Yes
SYSCVAVC	Vary activity found	Util
SYSAICVC	Appl% zAAP on CP	Util
SYSLOAVG	Load average	Util
SYSTCTVC	Total CPU time	Util
SYSUCTVC	Uncaptured time	Util
SYSCCTVC	Captured time	Util
SYSCUAVC	Average CPU Util% for zAAPs	No

Name	Description of the Variable	Report
SYSMUAVC	Average MVS Util% for zAAPs	No
SYSCUIVC	Average CPU Util% for zIIPs	No
SYSMUIVC	Average MVS Util% for zIIPs	No
SYSAHPVC	zAAP honor priority (YES/NO)	No
SYSIHPVC	zIIP honor priority (YES/NO)	No
SYSPKCVC	Average number of parked CPs	No
SYSPKAVC	Average number of parked zAAPs	No
SYSPKIVC	Average number of parked zIIPs	No

SYSRTD - Tabular report data table ERBRTDT3

RMF builds ERBRTDT3 when using SYSRTD as a report type.

Name	T	Description of the Variable	Report
RTDDTLLN	K	Logical line number	-
RTDDTPSN	K	Sequence number	-
RTDSYS	N	System identifier	Yes
RTDDAT	N	Data availability indication	Yes
RTDRTQ	N	Queued time / trx	Yes
RDTRTA	N	Active time / trx	Yes
RTDRTT	N	Total response time / trx	Yes
RTDTRAN	N	Ended transactions / second	Yes
RDTSSA	N	Transaction active percentage	Yes
RTDSSR	N	Transaction ready percentage	Yes
RTDSSD	N	Transaction delay percentage	Yes
RTDEXV	N	Execution velocity percentage	Yes
RTDEXD	N	Overall delay percentage	Yes

SYSSUM - Tabular report data table ERBSUMT3

RMF builds ERBSUMT3 when using SYSSUM as a report type.

Name	T	Description of the Variable	Report
SUMDTLLN	K	Logical line number	-
SUMDTPSN	K	Sequence number	-
SUMGRP	N	Group name	Yes
SUMTYP	N	Type of WLM group	Yes
SUMIMP	N	Importance of service class period	Yes
SUMVEG	N	Execution velocity goal	Yes
SUMEVA	N	Execution velocity actual	Yes
SUMRTGT	N	Response time goal	Yes
SUMRTGP	N	Response time goal percentile	Yes
SUMRTAT	N	Response time actual	Yes

Name	T	Description of the Variable	Report
SUMRTAP	N	Response time actual percentile	Yes
SUMPFID	N	Performance index	Yes
SUMTRAN	N	Ended transactions / second	Yes
SUMARTQ	N	Queued time	Yes
SUMARTA	N	Active time	Yes
SUMARTT	N	Total response time	Yes
SUMGOA	N	Goal type	Util
SUMDUR	N	Duration	Util
SUMRES	N	Resource group name	Util
SUMSMI	N	Service rate (capacity), min.	Util
SUMSMA	N	Service rate (capacity), max.	Util
SUMSRA	N	Service rate (capacity), actual	Util
SUMEGRP	N	Description of WLM group	Util

SYSWKM - Tabular report data table ERBWKMT3

RMF builds ERBWKMT3 when using SYSWKM as a report type.

Name	T	Description of the Variable	Report
WKMDTLLN	K	Logical line number	-
WKMDTPSN	K	Sequence number	-
WKMJOB	N	Jobname	Yes
WKMASI	N	Address space identification	Yes
WKMSYS	N	System identifier	Yes
WKMSRV	N	Service class name	Yes
WKMSER	N	Service percentage	Yes
WKMPRC	N	Processor using percentage	Yes
WKMVEL	N	Execution velocity percentage	Yes
WKMCAP	N	Capped delay percentage	Yes
WKMQUI	N	Address space quiesced percentage	Yes

WFEX - Tabular report data table ERBWFXT3

RMF builds ERBWFXT3 when using WFEX as a report type.

Name	T	Description of the Variable	Report
WFXDTLLN	K	Logical line number	-
WFXDTPSN	K	Sequence number	-
WFXATTR	N	Attribute	Util
WFXNAME	N	Name	Yes
WFXREASN	N	Reason	Yes
WFXDELAY	N	Delay	Yes
WFXPCAUS	N	Possible cause	Yes

XCF - Tabular report data table ERBXCFT3

RMF builds ERBXCFT3 when using XCF as a report type.

Name	T	Description of the Variable	Report
XCFDTLLN	K	Logical line number	-
XCFDTPSN	K	Sequence number	-
XCFPJOB	N	Jobname	Yes
XCFPCLA	N	Class (A, B, O, S, or T)	Yes
XCFPDMN	N	Domain number; no longer used	Yes
XCFPPGN	N	Performance group number; no longer used	Yes
XCFPSVCL	N	Service class name	Yes
XCFPODEL	N	Overall delay	Yes
XCF1SDEL	N	Delay percentage (Path 1)	Yes
XCF1PATH	N	Path 1	Yes
XCF2SDEL	N	Delay percentage (Path 2)	Yes
XCF2PATH	N	Path 2	Yes
XCF3SDEL	N	Delay percentage (Path 3)	Yes
XCF3PATH	N	Path 3	Yes
XCF4SDEL	N	Delay percentage (Path 4)	Yes
XCF4PATH	N	Path 4	Yes

ZFSSUM - Tabular report data table ERBZFST3

RMF builds ERBZFST3 when using ZFSSUM as a report type.

Name	T	Description of the Variable	Report
ZFSDTLLN	K	Logical line number	-
ZFSDTPSN	K	Sequence number	-
ZFSPAGG	N	Aggregate name	Report
ZFSPASIZ	N	Aggregate size	Report
ZFSPAUSE	N	Used percentage of aggregate	Report
ZFSPAMD1	N	Aggregate mode (R/W or R/O)	Report
ZFSPAMD2	N	Aggregate mode (CP or MS)	Report
ZFSPAFS	N	Number of filesystems in aggregate	Report
ZFSPARR	N	Read rate on aggregate	Report
ZFSPAWR	N	Write rate on aggregate	-

ZFSACT - Tabular report data table ERBZFAT3

RMF builds ERBZFAT3 when using ZFSACT as a report type.

Name	T	Description of the Variable	Report
ZFADTLLN	K	Logical line number	-
ZFADTPSN	K	Sequence number	-

Name	T	Description of the Variable	Report
ZFAPFILE	N	File system name, Mode, Quota limit, Quota usage percentage, Operation rate	Report
ZFAPFHKY	N	'F' - filesystem data line, 'M1' - mount point data line 1, 'M2' - mount point data line 2	Util
ZFAPFHAG	N	Aggregate name	Util
ZFAPFHNA	N	Filesystem name	Util
ZFAPFHMD	N	Filesystem mode	Util
ZFAPFHQL	N	Quota limit	Util
ZFAPFHQU	N	Quota usage percentage	Util
ZFAPFHOR	N	Operation rate	Util
ZFAPFHMP	N	Mount point	Util

End of Programming Interface information

Graphic report parameter table ERBPTGS3

Programming Interface information

The graphic report parameter table defines the layout of graphic reports for panel display and hardcopy printing. The first part describes general information about the graphic report. The second part describes information about labels per bar. The third part describes the column layout.

The format for general information is:

Name	T	Description of the Variable	Example
PTGREPNA	K	Report name (must be specified)	DEV HSM JES
PTGRHELP	N	Name for help panel – See name convention for HELP panels	
PTGRMINY	N	Length of Y-scale, if there is no bar exceeding this length. **1** for average number of user's time, **100** for percentage values	1 100
PTGRAXTI	N	Title of the axis • Percentage of Each User's Time • Percentage of The User's Time • Average Number of Active Users	1 100
PTGRSERU	N	Selection rule for bars: **0 :** One bar corresponds to one line **1 :** One bar corresponds to one line with the sequence number 1 **2 :** One bar corresponds to the summary of logical lines **3 :** Two bar-types result from all logical lines of a logical block • Bar type 1 corresponds to sequence number 1 • Bars of bar type 2 correspond to each line of the logical block	0 1 2 3 DELAY DEV, HSM, JES DEVR, ENQR STORR
PTGRBRNM	N	Number of bar types ‘1’ and ‘2’, represented by the character before the last character in the following variables.	1 2

The format for labels per bar is:

Variable Name	T	Variable Description	Example
PTGRLB10	N	Number of labels per bar for bar type 1	1 2
PTGRCL1 PTGRCL2	N N	ISPF COLUMN data-table variables containing the labels for bar-type 1.	
PTGRAP1 PTGRAP2	N N	Alpha part of the labels, which will be composed by this part and the last 4 digits of the data value.	'DMN', 'PG' in DELAY
PTGRLB20	N	Number of labels per bar for bar type 2	1 2
PTGRCL3 PTGRCL4 PTGRAP3 PTGRAP4	N N N N	(corresponding to PTGRCL1) (corresponding to PTGRCL2) (corresponding to PTGRAP1) (corresponding to PTGRAP2)	

The format for columns is:

Variable Name	T	Variable Description	Example
PTGRCPNM PTGRTV1 PTGRLD1 PTGRAL1 PTGRDC1	N N N N N	Number of data columns to be selected for the bar types. = number of color-pattern-text combin. (0, 1, 2, ... 9, represented by the last character of the variable. ISPF Column Table variable. This variable contains a specific data value of the tabular report after a TBGET to a row of the Data Column Table. (Must be specified) Legend ID, to specify a particular color-pattern-text combination of the Color-Pattern Table. The ID specifies the legend (color, pattern and subheader) for this data value. Transformation ID 0 : don't divide 1 : divide by 10 2 : divide by 100 bartype col ; If '0', the data value 0 : reports value in both bar types 1 : reports value in first bar type 2 : reports value in second bar type	0 1 ... 9 1 2 ... see color-pattern option table 0 2 0 1 2
PTGRTV2 PTGRLD2 PTGRAL2 PTGRDC2	N N N N	(corresponding to PTGRTV1) (corresponding to PTGRLD1) (corresponding to PTGRAL1) (corresponding to PTGRDC1)	
PTGRTV3 PTGRLD3 PTGRAL3 PTGRDC3	N N N N	(corresponding to ptgrtv1) (corresponding to PTGRLD1) (corresponding to PTGRAL1) (corresponding to PTGRDC1)	

Variable Name	T	Variable Description	Example
PTGRTV4 PTGRLD4 PTGRAL4 PTGRDC4	N N N N	(corresponding to PTGRTV1) (corresponding to PTGRLD1) (corresponding to PTGRAL1) (corresponding to PTGRDC1)	
PTGRTV5 PTGRLD5 PTGRAL5 PTGRDC5	N N N N	(corresponding to PTGRTV1) (corresponding to PTGRLD1) (corresponding to PTGRAL1) (corresponding to PTGRDC1)	
PTGRTV6 PTGRLD6 PTGRAL6 PTGRDC6	N N N N	(corresponding to PTGRTV1) (corresponding to PTGRLD1) (corresponding to PTGRAL1) (corresponding to PTGRDC1)	
PTGRTV7 PTGRLD7 PTGRAL7 PTGRDC7	N N N N	(corresponding to PTGRTV1) (corresponding to PTGRLD1) (corresponding to PTGRAL1) (corresponding to PTGRDC1)	
PTGRTV8 PTGRLD8 PTGRAL8 PTGRDC8	N N N N	(corresponding to PTGRTV1) (corresponding to PTGRLD1) (corresponding to PTGRAL1) (corresponding to PTGRDC1)	
PTGRTV9 PTGRLD9 PTGRAL9 PTGRDC9	N N N N	(corresponding to PTGRTV1) (corresponding to PTGRLD1) (corresponding to PTGRAL1) (corresponding to PTGRDC1)	
PTGRTV10 PTGRLD10 PTGRAL10 PTGRDC10	N N N N	(corresponding to PTGRTV1) (corresponding to PTGRLD1) (corresponding to PTGRAL1) (corresponding to PTGRDC1)	

End of Programming Interface information

RMF Phase driver table ERBPHDS3

Programming Interface information

The phase driver table has rows for each command and selection.

Variable Name	T	Variable Description
PHDREPNA	K	Name of the command or the long form of the report selection.
PHDREPSE	N	Selection string to be created. This string will be passed to the primary option panel to perform the command function.
PHDRPH1	N	Function to be performed for Phase 1. The string if not null, will be selected.
PHDRPH2	N	Function to be performed for Phase 2. The string if not null, will be selected.
PHDRPH3	N	Function to be performed for Phase 3. The string if not null, will be selected.
PHDRPH4	N	Function to be performed for Phase 4. The string if not null, will be selected.
PHDRET1	N	Return code passed from Phase 1. The Phase 2 and Phase 3 are executed only if the return code from this Phase is zero.
PHDRET2	N	Return code passed from Phase 2. The Phase 3 is executed only if the return code from this Phase is zero.
PHDRET3	N	Return code passed from Phase 3.
PHDRET4	N	Return code passed from Phase 5.
PHDRTAB1	N	Name of the ISPF table created by Phase 1. This table is input for Phase 2.
PHDRTAB2	N	Name of the ISPF table created by Phase 2. This table is input to Phase 3.

This table lists the report commands, selections, and the variables used for each phase (1,2,3,4). Phase 2 and 4 are null.

PHDREPNA	PHDREPSE	PHDRPH1	PHDRPH3	PHDRTAB1
CACHDET	S.9	PGM(ERB3RPH1) PARM(CACHDET)	PGM(ERB3RDSP)	ERBCADT3
CACHSUM	S.8	PGM(ERB3RPH1) PARM(CACHSUM)	PGM(ERB3RDSP)	ERBCAST3
CFACT	S.7	PGM(ERB3RPH1) PARM(CFACT)	PGM(ERB3RDSP)	ERBCFAT3
CFOVER	S.5	PGM(ERB3RPH1) PARM(CFOVER)	PGM(ERB3RDSP)	ERBCFOT3
CFSYS	S.6	PGM(ERB3RPH1) PARM(CFSYS)	PGM(ERB3RDSP)	ERBCFST3
CHANNEL	3.12	PGM(ERB3RPH1) PARM(CHANNEL)	PGM(ERB3RDSP)	ERBCHAT3
CPC	1.3	PGM(ERB3RPH1) PARM(CPC)	PGM(ERB3RDSP)	ERBCPCT3
DELAY	1.4	PGM(ERB3RPH1) PARM(DELAY)	PGM(ERB3RDSP)	ERBJDET3
DEV	3.2	PGM(ERB3RPH1) PARM(DEV)	PGM(ERB3RDSP)	ERBDEVT3
DEVR	3.3	PGM(ERB3RPH1) PARM(DEVR)	PGM(ERB3RDSP)	ERBDVRT3
DSINDEX	S.D	PGM(ERB3RHP1) PARM(DSINDEX)	PGM(ERB3RDSP)	ERBDSIT3
DSND	3.3A	PGM(ERB3RPH1) PARM(DSND)	PGM(ERB3RDSP)	ERBDNDT3
DSNJ	2.1A	PGM(ERB3RPH1) PARM(DSNJ)	PGM(ERB3RDSP)	ERBDNJT3
DSNV	3.3B	PGM(ERB3RPH1) PARM(DSNV)	PGM(ERB3RDSP)	ERBDNVT3

PHDREPNA	PHDREPSE	PHDRPH1	PHDRPH3	PHDRTAB1
ENCLAVE	1.6	PGM(ERB3RPH1) PARM(ENCLAVE)	PGM(ERB3RDSP)	ERBENCT3
ENQ	3.4	PGM(ERB3RPH1) PARM(ENQ)	PGM(ERB3RDSP)	ERBENQT3
ENQR	3.5	PGM(ERB3RPH1) PARM(ENQR)	PGM(ERB3RDSP)	ERBEQRT3
GROUP	1.5	PGM(ERB3RPH1) PARM(GROUP)	PGM(ERB3RDSP)	ERBGRTT3
HSM	4.1	PGM(ERB3RPH1) PARM(HSM)	PGM(ERB3RDSP)	ERBHSMT3
IOQ	3.13	PGM(ERB3RPH1) PARM(IOQ)	PGM(ERB3RDSP)	ERBIOQT3
JES	4.2	PGM(ERB3RPH1) PARM(JES)	PGM(ERB3RDSP)	ERBJEST3
JOB	2.5	PGM(ERB3RPH1) PARM(JOB)	PGM(ERB3RDSP)	ERBJDJT3
OPD	1.7	PGM(ERB3RPH1) PARM(OPD)	PGM(ERB3RDSP)	ERBOPDT3
PROC	3.1	PGM(ERB3RPH1) PARM(PROC)	PGM(ERB3RDSP)	ERBPRCT3
RLSDS	S.11	PGM(ERB3RPH1) PARM(RLSDS)	PGM(ERB3RDSP)	ERBVRDT3
RLSLRU	S.12	PGM(ERB3RPH1) PARM(RLSLRU)	PGM(ERB3RDSP)	ERBVRLT3
RLSSC	S.10	PGM(ERB3RPH1) PARM(RLSSC)	PGM(ERB3RDSP)	ERBVRST3
STOR	3.6	PGM(ERB3RPH1) PARM(STOR)	PGM(ERB3RDSP)	ERBSTRT3
STORC	3.10	PGM(ERB3RPH1) PARM(STORC)	PGM(ERB3RDSP)	ERBCSUT3
STORCR	3.11	PGM(ERB3RPH1) PARM(STORCR)	PGM(ERB3RDSP)	ERBCRST3
STORF	3.7	PGM(ERB3RPH1) PARM(STORF)	PGM(ERB3RDSP)	ERBSTFT3
STORR	3.8	PGM(ERB3RPH1) PARM(STORR)	PGM(ERB3RDSP)	ERBSRRT3
STORS	3.9	PGM(ERB3RPH1) PARM(STORS)	PGM(ERB3RDSP)	ERBSRST3
SYSENQ	S.4	PGM(ERB3RPH1) PARM(SYSENQ)	PGM(ERB3RDSP)	ERBEQST3
SYSINFO	1.2	PGM(ERB3RPH1) PARM(SYSINFO)	PGM(ERB3RDSP)	ERBSYST3
SYSRTD	S.2	PGM(ERB3RPH1) PARM(SYSRTD)	PGM(ERB3RDSP)	ERBRTDT3
SYSSUM	S.1	PGM(ERB3RPH1) PARM(SYSSUM)	PGM(ERB3RDSP)	ERBSUMT3
SYSWKM	S.3	PGM(ERB3RPH1) PARM(SYSWKM)	PGM(ERB3RDSP)	ERBWKMT3
WFEX	1.1	PGM(ERB3RPH1) PARM(WFEX)	PGM(ERB3RDSP)	ERBWFXT3
XCF	4.3	PGM(ERB3RPH1) PARM(XCF)	PGM(ERB3RDSP)	ERBXCFT3

End of Programming Interface information

Chapter 9. Diagnosing problems in RMF

Before using the information provided in this topic, you should:

- Be familiar with RMF. See *z/OS RMF User's Guide* for specific information about RMF sessions.
- Understand the information in the *z/OS Problem Management*.

Identifying problems

The following table lists each problem type and its meaning. The table directs you to a diagnostic procedure for that problem type or to another book.

Table 17. Problem Types

For these problem types:	See the following:
Abend X'0D5': RMF detected an address space identifier (ASID) that was not valid.	"Diagnosing abend 0D5" on page 318
Abend X'0FE': An error occurred, ending the Monitor I ZZ session.	"Diagnosing abend 0FE" on page 319
Abend unexpected by RMF: A module with the prefix ERB abnormally ended with a system completion code other than the codes listed above.	"Diagnosing an abend unexpected by RMF" on page 320
Message with ERB prefix.	"Diagnosing a message with an ERB prefix" on page 321
Message with CEE or EDC prefix.	"Diagnosing a message with a CEE/EDC prefix" on page 321
Abend with a user completion code issued by a module with an ERB prefix.	*z/OS RMF Messages and Codes*
Incorrect output: Unusually large or small numbers appeared in an RMF report.	"Diagnosing incorrect output" on page 323
Documentation error: An error in RMF documentation was detected.	"Diagnosing a documentation error" on page 323
Empty Monitor III JES delays report: A Monitor III JES delays report was displayed that contained no data. The headings appeared, but the fields were blank.	"Diagnosing an empty Monitor III JES Delays report" on page 324

Diagnosing abend 0D5

Use this procedure when a module with the prefix ERB abnormally ends with the system completion code X'0D5'. RMF detected an ASID that was not valid.

Table 18. Diagnostic Procedure for Abend 0D5

Diagnostic Procedure	References
1. Obtain the SYS1.LOGREC error record and format it with EREP to obtain a detail edit report.	*EREP User's Guide* for EREP formatting *z/OS MVS Diagnosis: Tools and Service Aids* to read the SYS1.LOGREC record *z/OS MVS Diagnosis: Tools and Service Aids*
2. Check the SYS1.LOGREC error record to determine if the abend occurred in module ERBMFPVS. If the abend did not occur in module ERBMFPVS, continue with step 4. Otherwise proceed with the next step.	*z/OS MVS Diagnosis: Tools and Service Aids* to read the SYS1.LOGREC record *z/OS MVS Diagnosis: Tools and Service Aids*
3. Obtain the SVC dump for the abend.	*z/OS Problem Management*
4. Format the dump with the IPCS VERBEXIT LOGDATA subcommand to see the system diagnostic work area (SDWA) and the search argument. In the dump, find the offset of the failing instruction into the module. If analyzing the dump at a terminal, use the IPCS WHERE subcommand to locate the failing instruction and to find its offset into the module.	*z/OS MVS IPCS Commands* for the VERBEXIT LOGDATA and WHERE subcommands
5. Determine if the ASID is not valid. Either: • The address space does not exist • The address space is swapped out. Correct the problem and reinitialize RMF. If you cannot determine why the ASID is not valid, continue with the next step.	
6. Develop a search argument consisting of: • Programmer identifier: IDS/566527404 • System abend code: AB/S00D5 • CSECT name: RIDS/cccccccc • Load module name: RIDS/cccccccc#L Use the search argument to search problem reporting data bases. If the search finds that the problem has been reported before, request the problem fix. If not, continue with the next step.	Developing a search argument for RMF *z/OS Problem Management*
7. Report the problem to IBM. Provide the following problem data: • Load module name and level • CSECT name and level • Offset of the failing instruction into the module • SVC dump (softcopy) • Search argument • Current RMF Monitor I options • RMF version and release • Name and level of the operating system with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service.	Reporting a problem to IBM

Diagnosing abend 0FE

Use this procedure when RMF ends with a system completion code X'0FE'. RMF detected an error while sampling data about the state of the system.

Table 19. Diagnostic Procedure for Abend 0FE

Diagnostic Procedure	References
1. Obtain the SYS1.LOGREC error record and format it with EREP to obtain a detail edit report.	*EREP User's Guide* for EREP formatting *z/OS MVS Diagnosis: Tools and Service Aids* to read the SYS1.LOGREC record *z/OS MVS Diagnosis: Tools and Service Aids*
2. Locate the 0FE entry in the SYS1.LOGREC error record. The entry preceding X'0FE' indicates the module that abended.	*z/OS MVS Diagnosis: Tools and Service Aids*
3. Set a SLIP trap for the abend code preceding the X'0FE' entry in the SYS1.LOGREC error record. Request the SLIP to produce an SVC dump. If you cannot reproduce the situation, continue with step 5.	*z/OS Problem Management*
4. Format the dump produced by the SLIP trap with the IPCS VERBEXIT LOGDATA subcommand and keep as softcopy. In the dump, find the offset of the failing instruction into the module. If analyzing the dump at a terminal, use the IPCS WHERE subcommand to locate the failing instruction and to find its offset into the module.	*z/OS MVS IPCS Commands* for the VERBEXIT LOGDATA and WHERE subcommands
5. Develop a search argument consisting of: • Program identifier: PIDS/566527404 • Load module name: RIDS/cccccccc#L • CSECT name: RIDS/cccccccc • System abend code: AB/S00FE • SLIP trap abend code: AB/S0xxx Use the search argument to search problem reporting data bases. If the search finds that the problem has been reported before, request the problem fix. If not, continue with the next step.	Developing a search argument for RMF *z/OS Problem Management*
6. Report the problem to IBM. Provide the following problem data: • Load module name and level • CSECT name and level • Offset of the failing instruction into the module • Registers at time of abend • SVC dump produced by the SLIP trap (softcopy) • SYS1.LOGREC (softcopy) • Search argument • Current RMF Monitor I options • RMF version and release • Console log of this situation including related ERBxxxI messages • Name and level of the operating system with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service.	Reporting a problem to IBM

Diagnosing an abend unexpected by RMF

Use this procedure when a module with the prefix ERB abnormally ends with a system completion code other than the codes listed in the table under "Identifying problems" on page 317.

Table 20. Diagnostic Procedure for an Abend Unexpected by RMF

Diagnostic Procedure	References
1. Look at the explanation for the abend code and any accompanying reason code. Take the recommended actions.	*z/OS MVS System Codes* for an explanation of the abend code
2. Obtain messages accompanying the abend. Look at their explanations and take any recommended actions.	Please visit the LookAt home page at http://www.ibm.com/systems/z/os/zos/bkserv/lookat/ for getting the fastest message explanation.
3. Obtain the SYS1.LOGREC error record, the dump, or both for the abend (softcopy).	*z/OS MVS Diagnosis: Tools and Service Aids*
4. Format the dump or SYS1.LOGREC record to see the SDWA and the search argument: • Format an SVC dump or SYSMDUMP ABEND dump with the IPCS VERBEXIT LOGDATA subcommand. • Format a SYS1.LOGREC record with EREP to obtain a detail edit report. Find the heading VARIABLE RECORDING AREA (SDWAVRA). Note the SDWAVRA keys, lengths, and contents.	*z/OS MVS IPCS Commands* for the VERBEXIT LOGDATA subcommand *EREP User's Guide* for EREP formatting *z/OS MVS Diagnosis: Tools and Service Aids* *z/OS MVS Diagnosis: Tools and Service Aids*
5. In the dump, find the offset of the failing instruction within the CSECT of the load module. If analyzing the dump at a terminal, use the IPCS WHERE subcommand to locate the failing instruction and to find its offset into the module.	*z/OS MVS IPCS Commands* for the WHERE subcommand
6. Use the WRITELOG system command to print the system log (keep as softcopy). The system log shows all system messages and commands issued. Make sure you print the log for the time period when the abend occurred.	*z/OS MVS System Commands* for the WRITELOG command
7. Develop a search argument consisting of: • Program identifier: PIDS/566527404 • System abend code: AB/S0xx • Abend reason code: PRCS/xxxxxxxx • Message identifier: MS/cccnnns • CSECT: RIDS/cccccccc • Load module name: RIDS/cccccccc#L Use the search argument to search problem reporting data bases. If the search finds that the problem has been reported before, request the problem fix. If not, continue with the next step.	Developing a search argument for RMF *z/OS Problem Management*

Table 20. Diagnostic Procedure for an Abend Unexpected by RMF (continued)

Diagnostic Procedure	References
8. Report the problem to IBM. Provide the following problem data: • Accompanying messages • The dump, SYS1.LOGREC error record, or both (softcopy) • SDWAVRA keys, lengths, and contents • Hardcopy (or better softcopy) of console log • Search argument • Offset of the failing instruction into the module • Current RMF options for session running • RMF version and release • Load module name and level • CSECT name and level • Name and level of the operating system with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service.	Reporting a problem to IBM

Diagnosing a message with an ERB prefix

Use this procedure when you receive a message with an ERB prefix.

Table 21. Diagnostic Procedure for a Message with an ERB Prefix

Diagnostic Procedure	References
1. Determine which session was running when the message was issued. • If the message was issued within an RMF ISPF reporter session, use the help facility RMF supplies for an explanation of the message and user response. • If the message was received on the main operator console, it is a system message. If you need assistance with a message, continue with the next step.	Press the HELP key (PF1) for an explanation of the message. These message explanations are also listed in the *z/OS RMF Messages and Codes* book. For an explanation and an appropriate operator response for system messages, please visit the LookAt home page at http://www.ibm.com/systems/z/os/zos/bkserv/lookat/ .
2. Report the problem to IBM. Provide the following problem data: • Message number • RMF version and release • RMF session: Monitor I, II, III or Postprocessor • Name and level of the operating system with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service.	Reporting a problem to IBM

Diagnosing a message with a CEE/EDC prefix

Use this procedure when you receive a message with a CEE or EDC prefix.

These message are created by integrated LE/370 routines during a Postprocessor session.

Table 22. Diagnostic Procedure for a Message with a CEE/EDC Prefix

Diagnostic Procedure	**References**
1. Increase the region size for the Postprocessor job. If the problem is not solved, continue with the next step.	See the *z/OS RMF User's Guide* for details. See *z/OS Language Environment Debugging Guide* for an explanation of the message.
2. Report the problem to IBM. Provide the following problem data: • Message number • RMF version and release • RMF session: Postprocessor • Name and level of the operating system with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service.	Reporting a problem to IBM

Diagnosing incorrect output

Use this procedure when unusually large or small numbers appear in any RMF reports. Inaccurate input might have been given to RMF from another system component.

Table 23. Diagnostic Procedure for Incorrect Output

Diagnostic Procedure	References
1. Obtain a hardcopy of the report that has unusual numbers. If the problem is a Monitor III JES delays report which has no data, see “Diagnosing an empty Monitor III JES Delays report” on page 324.	*z/OS RMF User's Guide*
2. If you are running a Monitor I or Monitor II session or the Postprocessor, and you are collecting SMF data, print the contents of the SMF records which generated *this* report (and keep them as softcopy). If you are running a Monitor III session, keep any VSAM data sets used to hold data during the session. If the problem is with a report produced by a Monitor III SMF record, print the contents of that SMF record (and keep it as softcopy).	*z/OS RMF User's Guide*
3. Report the problem to IBM. Provide the following problem data: • Hardcopy (or better softcopy) of the report • Accompanying messages • RMF version and release • RMF session: Monitor I, II, III or Postprocessor • SMF record contents • VSAM data set contents if running a Monitor III session • Current RMF options • Name and level of the operating system with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service.	Reporting a problem to IBM

Diagnosing a documentation error

Use this procedure when you find an error in RMF documentation.

Table 24. Diagnostic Procedure for a Documentation Error

Diagnostic Procedure	References
1. If you have a problem with an RMF publication use the Reader's Comment Form of that book to report the documentation error. Be specific when reporting the error. If there is no Reader's Comment Form, send your description of the documentation error to the address listed in the edition notice of the book. Include the following information: • Publication title • Publication order number • Page number containing the problem • Thorough description of the problem. If the problem is with a form of documentation other than a publication, continue with the next step.	See the Reader's Comment Form of this book for an example.

Table 24. Diagnostic Procedure for a Documentation Error (continued)

Diagnostic Procedure	References
2. If the problem was with an RMF ISPF panel, type PANELID on the command line to display the panel ID. Use the Reader's Comment Form of *z/OS RMF Report Analysis* to report the problem. Include the following information: • RMF version and release • Panel ID • Thorough description of the problem.	See the Reader's Comment Form of this book for an example.
3. Report the problem to the IBM Support Center *only* in the following situations: • The correction to the documentation is needed to prevent a severe problem. • You are not sure if the problem is a documentation error or product error. If you report the problem to the IBM Support Center, provide the following data: • RMF version and release • The name and order number of the publication you are using and the page containing the error • If the error was on an online panel, provide the panel ID. Type PANELID on the command line to display the panel ID. • Name and level of the operating system with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service.	Reporting a problem to IBM

Diagnosing an empty Monitor III JES Delays report

Use this procedure when no data appears in a Monitor III JES delays report.

If there are no jobs delayed by JES, RMF generates an empty Monitor III JES delays report. In this case, an empty report is not considered a problem.

Table 25. Diagnostic Procedure for Empty Monitor III JES Delays Report

Diagnostic Procedure	References
1. Find out if the JES2 or JES3 control blocks have changed at your installation. If so, Monitor III modules might be affected.	For the RMF/JES interface module installation procedures, see *z/OS Program Directory*
2. If the JES control blocks have changed, you must reassemble and link-edit the RMF source modules that map offsets to the JES control block fields sampled by Monitor III using the new JES macros. If the problem occurs again, continue with the next step.	
3. Report the problem to IBM. Provide the following problem data: • Accompanying messages • RMF version and release • Installed JES level • Hardcopy of the report • VSAM data set contents • Name and level of the operating system with a list of program temporary fixes (PTF) applied at the time of the problem and all installation modifications, exits, and products with other than Class A service.	Reporting a problem to IBM

Obtaining a dump from Monitor II or Monitor III

If an error occurs in the Monitor II or Monitor III reporter session, RMF prompts you whether you want to write a dump. Follow the steps of the diagnostic procedure to obtain a dump:

1. Enter into the command line:

   ```
   TSO FREE FI(SYSUDUMP SYSABEND)
   ```

   You can ignore messages, for example: `IKJ56247I FILE xxxxxxxx NOT FREED, IS NOT ALLOCATED`
2. Enter into the command line:

   ```
   TSO ALLOC FI(SYSMDUMP) DA(dsname) NEW SP(200 200) CYL REUSE REL
   ```

   If the command does not fit into the command line, start the split-screen mode to enter the command.
3. Answer `Y` in the dump request panel:

   `Would you like a dump?  Enter Y or N. ===>` **Y**
4. The system now writes an unformatted dump to the data set just allocated. This may take some time. When it is finished, the system issues message:

   IEA993I SYSMDUMP TAKEN TO dsname
5. You can now process the dump with IPCS.

Developing a search argument for RMF

You or IBM can use a search argument to search a problem reporting data base to look for a problem similar to the one you encountered. If the problem was reported previously, a problem reporting data base contains information about the problem and, possibly, a fix. See *z/OS Problem Management* for detailed descriptions of formatting search arguments and searching data bases.

The following table shows symptoms for RMF search arguments. The table summarizes the symptoms recommended in the diagnostic procedures in Chapter 9, "Diagnosing problems in RMF."

Use the **free-format** if your installation has a free-format search tool, such as INFO/System with the INFO/MVS data base.

Use the **structured** format if your installation has a structured format search tool, such as INFO/Management or INFO/Access.

Table 26. Search Arguments from Diagnostic Procedures

Description	Free Format	Structured Format	Examples
System abend code	abendhhh	AB/S0hhh	abend0D5 AB/S00D5
User completion code	abenddddd	AB/Udddd	abend1207 AB/U1207
Message identifier	msgccccccc	MS/ccccccc	msgERB251I MS/ERB251I
Load module name	cccccccc	RIDS/cccccccc#L	ERB3GMFC RIDS/ERB3GMFC#L
CSECT name (object module)	cccccccc	RIDS/cccccccc	ERB3GISS RIDS/ERB3GISS
Return code	rchhhhhhhh	PRCS/hhhhhhhh	00000020 PRCS/00000020
Program identifier The first four characters identify the product MVS and the last five characters identify RMF.	ccccccccc	PIDS/ccccccccc	566527404 PIDS/566527404

Reporting a problem to IBM

The following tables identify the information you need to collect before calling IBM to report an RMF problem. When you report a problem, you need to describe your system and the problem you experienced. The IBM Support Center personnel uses this information to see if the problem is already known to IBM, check whether a fix is available, or determine how to correct the problem.

Table 27 lists the problem data you need to collect before calling IBM to report a problem with an abend X'0D5'.

Table 27. Checklist for Reporting a Problem with an Abend 0D5

Problem data	**Example or reference**
Load module name and level[1]	ERB*xxxxx* U*nnnnnn*[2]
CSECT name and level	ERB*xxxxx* U*nnnnnn*
Offset of the failing instruction into the module	X'090AF'
SVC dump (SYSMDUMP as softcopy) *z/OS Problem Management*	
Search argument	"Developing a search argument for RMF" on page 326 *z/OS Problem Management*
Current RMF options	Use the RMF DISPLAY command to display the current options on the operator console. *z/OS RMF User's Guide*
RMF version and release	z/OS V1R13 RMF
Program temporary fix (PTF) numbers	U*nnnnnn*
Other problem data developed while using the *z/OS Problem Management* book and the diagnostic procedures in Chapter 1	

Note:
1. The level can be U*nnnnnn*, HRM*nnnn*, or JRM*nnnn*.
2. *xxxxx* and *nnnnnn* are just placeholders. The actual data can be obtained from the dump.

Reporting a problem

Table 28 lists the problem data you need to collect before calling IBM to report a problem with an abend X'0FE'.

Table 28. Checklist for Reporting a Problem with an Abend 0FE

Problem data	**Example or reference**
Load module name and level[1]	ERB*xxxxx* U*nnnnnn*[2]
CSECT name and level	ERB*xxxxx* U*nnnnnn*
Offset of the failing instruction into the module	X'090AF'
SVC dump produced by the SLIP trap (softcopy)	*z/OS Problem Management*
SYS1.LOGREC (softcopy)	*z/OS MVS Diagnosis: Tools and Service Aids*
Console log (softcopy)	
Search argument	"Developing a search argument for RMF" on page 326 *z/OS Problem Management*
Current Monitor I options	Use the RMF DISPLAY command to display the current options on the operator console. *z/OS RMF User's Guide*
RMF version and release	z/OS V1R13 RMF
Program temporary fix (PTF) numbers	U*nnnnnn*
Other problem data developed while using the *z/OS Problem Management* book and the diagnostic procedures in Chapter 1	

Note:
1. The level can be U*nnnnnn*, HRM*nnnn*, or JRM*nnnn*.
2. *xxxxx* and *nnnnnn* are just placeholders. The actual data can be obtained from the dump.

Table 29 lists the problem data you need to collect before calling IBM to report a problem with an unexpected abend.

Table 29. Checklist for Reporting a Problem with an Unexpected Abend

Problem data	**Example or reference**
Load module name and level[1]	ERB*xxxxx* U*nnnnnn*[2]
CSECT name and level	ERB*xxxxx* U*nnnnnn*
Messages accompanying the problem, including the message identifier and variable data in the message text	Identifier: ERB259I Text: EXCEPTION REPORTING TERMINATED *z/OS RMF Messages and Codes* Please visit the LookAt home page at http://www.ibm.com/systems/z/os/zos/bkserv/lookat/ for getting the fastest message explanation.
SVC dump (SYSMDUMP as softcopy)	*z/OS Problem Management*
SYS1.LOGREC error record (softcopy)	*z/OS Problem Management*
SDWAVRA keys, lengths, and contents	*z/OS Problem Management*
System log to show system messages and commands (softcopy)	*z/OS MVS System Commands* for WRITELOG command
Search argument	"Developing a search argument for RMF" on page 326 *z/OS Problem Management<*
In case of a Monitor III error, provide the contents of the VSAM data set belonging to this problem	
Offset of the failing instruction into the module	X'090AF'
Current RMF options for session running	Use the RMF DISPLAY command to display the current options on the operator console. *z/OS RMF User's Guide*
RMF version and release	z/OS V1R13 RMF
Program temporary fix (PTF) numbers	U*nnnnnn*
Other problem data developed while using the *z/OS Problem Management* book and the diagnostic procedures in Chapter 1	

Note:
1. The level can be U*nnnnnn*, HRM*nnnn*, or JRM*nnnn*.
2. *xxxxx* and *nnnnnn* are just placeholders. The actual data can be obtained from the dump.

Reporting a problem

Table 30 lists the problem data you need to collect before calling IBM to report a problem with an ERB or CEE/EDC message.

Table 30. Checklist for Reporting a Problem with an ERB CEE/EDC Message

Problem data	Example or reference
Message number	ERB671I
RMF version and release	z/OS V1R13 RMF
RMF session	Monitor I, II, III or Postprocessor
Program temporary fix (PTF) numbers	U*nnnnnn*[1]
Other problem data developed while using the *z/OS Problem Management* book and the diagnostic procedures in Chapter 1.	
Note: 1. *nnnnnn* is just a placeholder. The actual data can be obtained from the dump.	

Table 31 lists the problem data you need to collect before calling IBM to report a problem with incorrect output.

Table 31. Checklist for Reporting a Problem with Incorrect Output

Problem data	Example or reference
Hardcopy of report	*z/OS RMF User's Guide*
Accompanying messages	*z/OS RMF Messages and Codes* Please visit the LookAt home page at http://www.ibm.com/systems/z/os/zos/bkserv/lookat/ for getting the fastest message explanation.
RMF version and release	z/OS V1R13 RMF
RMF session	Monitor I, II, III or Postprocessor
SMF record contents	*z/OS RMF User's Guide*
Unloaded VSAM data set contents if running a Monitor III session	*z/OS RMF User's Guide*
Current RMF options	Copy from SYS1.PARMLIB
Program temporary fix (PTF) numbers	U*nnnnnn*[1]
Other problem data developed while using the *z/OS Problem Management* book and the diagnostic procedures in Chapter 1	
Note: 1. *nnnnnn* is just a placeholder. The actual data can be obtained from the dump.	

Table 32 lists the problem data you need to collect before calling IBM to report a problem with documentation.

Table 32. Checklist for Reporting a Documentation Error

Problem data	Example or reference
Publication title	*z/OS RMF Messages and Codes*
Publication order number	SC33-7993
Page number containing problem	3-45
Panel ID	ERB3PRM
RMF version and release	z/OS V1R13 RMF
Program temporary fix (PTF) numbers	U*nnnnnn*[1]

Table 32. Checklist for Reporting a Documentation Error (continued)

Problem data	Example or reference
Thorough description of the problem.	
Other problem data developed while using the *z/OS Problem Management* book and the diagnostic procedures in Chapter 1	
Note: 1. *nnnnnn* is just a placeholder. The actual data can be obtained from the dump.	

Table 33 lists the problem data you need to collect before calling IBM to report a problem with a Monitor III JES delays report.

Table 33. Checklist for Reporting a Problem with a Monitor III JES Delays Report

Problem data	Example or reference
Accompanying messages	*z/OS RMF Messages and Codes* Please visit the LookAt home page at http://www.ibm.com/systems/z/os/zos/bkserv/lookat/ for getting the fastest message explanation.
Installed JES level	JES3 Release x.x
Hardcopy of report	*z/OS RMF User's Guide*
Unloaded VSAM data set contents	*z/OS RMF User's Guide*
RMF version and release	z/OS V1R13 RMF
Program temporary fix (PTF) numbers	U*nnnnnn*[1]
Other problem data developed while using the *z/OS Problem Management* book and the diagnostic procedures in Chapter 1	
Note: 1. *nnnnnn* is just a placeholder. The actual data can be obtained from the dump.	

Accessibility

Publications for this product are offered in Adobe Portable Document Format (PDF) and should be compliant with accessibility standards. If you experience difficulties when using PDF files, you may view the information through the z/OS Internet Library website or the z/OS Information Center. If you continue to experience problems, send an email to mhvrcfs@us.ibm.com or write to:

IBM Corporation
Attention: MHVRCFS Reader Comments
Department H6MA, Building 707
2455 South Road
Poughkeepsie, NY 12601-5400
U.S.A.

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS® enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

z/OS information is accessible using screen readers with the BookServer or Library Server versions of z/OS books in the Internet library at:

`http://www.ibm.com/systems/z/os/zos/bkserv/`

Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users accessing the Information Center using a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line, because they can be considered as a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, you know that your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is given the format 3 * FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol giving information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this indicates a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should refer to separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- ? means an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

- ! means a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicates that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines 2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP will be applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE), the default option KEEP only applies to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.
- * means a syntax element that can be repeated 0 or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

  Notes:

  1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
  2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.
  3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.
- + means a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times; that is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the * symbol, the + symbol can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.

Glossary

This glossary contains chiefly definitions of terms used in this book, but some more general RMF and MVS terms are also defined.

Words that are set in *italics* in the definitions are terms that are themselves defined in the glossary.

A

APPC/MVS. Advanced program-to-program communication

ASCH address space. APPC transaction scheduler address space

AS. *Address space*

address space. That part of MVS main storage that is allocated to a job.

auxiliary storage (AUX). All addressable storage, other than main storage, that can be accessed by means of an I/O channel; for example storage on direct access devices.

B

background session. In RMF, a monitor session that is started and controlled from the operator console. Contrast with *interactive session*

balanced systems. To avoid bottlenecks, the system resources (CP, I/O, storage) need to be balanced.

basic mode. A central processor mode that does not use logical partitioning. Contrast with *logically partitioned (LPAR) mode*.

bottleneck. A system resource that is unable to process work at the rate it comes in, thus creating a queue.

C

callable services. Parts of a program product that have a published external interface and can be used by application programs to interact with the product.

captured storage. See shared page group.

capture ratio. The ratio of reported CPU time to total used CPU time.

central processor (CP). The part of the computer that contains the sequencing and processing facilities for instruction execution, initial program load, and other machine operations.

central processor complex (CPC). A physical collection of hardware that consists of central storage, one or more central processors, timers, and channels.

channel path. The channel path is the physical interface that connects control units and devices to the CPU.

CICS. Customer Information Control System

CIM provider. A CIM provider is the link between the CIM server and the system interfaces. It allows the CIM server to access and manage the resources. Each CIM provider exposes the resources it represents in a standard way, using a small number of classes from the CIM schema or derived from the CIM schema. RMF monitoring providers are CIM providers implemented by RMF.

contention. Two or more incompatible requests for the same resource. For example, contention occurs if a user requests a resource and specifies exclusive use, and another user requests the same resource, but specifies shared use.

coupling facility. See *Cross-system Extended Services/Coupling Facility*.

CP. *Central processor*

criteria. Performance criteria set in the WFEX report options. You can set criteria for all report classes (PROC, SYSTEM, TSO, and so on).

CPU speed. Measurement of how much work your CPU can do in a certain amount of time.

cross-system coupling facility (XCF). A component of MVS that provides functions to support cooperation between authorized programs running within a *sysplex*.

Cross-system Extended Services/Coupling Facility (XES/CF). Provides services for MVS systems in a sysplex to share data on a coupling facility (CF).

CS. Central storage

Customer Information Control System (CICS). An IBM licensed program that enables transactions entered at remote terminals to be processed concurrently by user-written application programs. It includes facilities for building, using, and maintaining data bases.

cycle. In RMF, the time at the end of which one sample is taken. Varies between 50 ms and 9999 ms. See also *sample*.

D

data sample. See *sample*

DCM. See *Dynamic Channel Path Management*

delay. The delay of an address space represents a job that needs one or more resources but that must wait because it is contending for the resource(s) with other users in the system.

direct access storage device (DASD). A device in which the access time is effectively independent of the location of the data. Usually: a magnetic disk device.

DLY. Delay

DP. Dispatching priority

dynamic channel path management. Dynamic channel path management provides the capability to dynamically assign channels to control units in order to respond to peaks in demand for I/O channel bandwidth. This is possible by allowing you to define pools of so-called floating channels that are not related to a specific control unit. With the help of the Workload Manager, channels can float between control units to best service the work according to their goals and their importance.

E

EMIF. ESCON multiple image facility

enclave. An enclave is a group of associated dispatchable units. More specifically, an enclave is a group of SRB routines that are to be managed and reported on as an entity.

EPDM. Enterprise Performance Data Manager/MVS

ES. Expanded storage

ESCON multiple image facility (EMIF). A facility that allows channels to be shared among PR/SM logical partitions in an ESCON environment.

execution velocity. A measure of how fast work should run when ready, without being delayed for processor or storage access.

exception reporting. In RMF, the reporting of performance measurements that do not meet user-defined criteria. Shows potential performance problems explicitly, thus avoiding the need for constant monitoring.

expanded storage (ES). (1) An extension of processor storage. (2) Optional high-speed storage that transfers 4KB pages to and from central storage.

G

generalized trace facility (GTF). A service program that records significant system events, such as supervisor calls and start I/O operations, for the purpose of problem determination.

GO mode. In RMF, the Monitor III mode in which the screen is updated with the interval you specified in your session options. The terminal cannot be used for anything else when it is in GO mode. See also *mode*.

graphic mode. In RMF Monitor III, the mode which presents the performance data from the system in graphic format using the GDDM product. Contrast with *tabular mode*.

GTF. generalized trace facility

H

high-speed buffer (HSB). A cache or a set of logically partitioned blocks that provides significantly faster access to instructions and data than provided by central storage.

HS. hiperspace

HSB. High-speed buffer

HSM. Hierarchical Storage Manager

I

IBM System z9 Application Assist Processor (zAAP). A special purpose processor configured for running Java programming on selected zSeries machines.

IBM System z9 Integrated Information Processor (zIIP). A special purpose processor designed to help free-up general computing capacity and lower overall total cost of computing for selected data and transaction processing workloads for business intelligence (BI), ERP and CRM, and selected network encryption workloads on the mainframe.

IMS. Information Management System

Information Management System (IMS). A database/data communication (DB/DC) system that can manage complex databases and networks. Synonymous with IMS/VS.

interactive session. In RMF, a monitor display-session that is controlled from the display terminal. Contrast with *background session*.

J

JES. Job Entry Subsystem

L

LCU. Logical control unit. Logical control units are also called 'Control Unit Headers ' (CUH). For details about LCU/CUH please refer to the applicable *System z Input/Output Configuration Program User's Guide for ICP IOCP* (SB10-7037).

logically partitioned (LPAR) mode. A central processor mode that is available on the Configuration frame when using the PR/SM feature. It allows an operator to allocate processor unit hardware resources among logical partitions. Contrast with *basic mode*.

logical partition (LP). A subset of the processor hardware that is defined to support an operating system. See also *logically partitioned (LPAR) mode*.

LP. Logical partition

LPAR. Logically partitioned (mode)

LPAR cluster. An LPAR cluster is the subset of the systems that are running as LPARs on the same CEC. Based on business goals, WLM can direct PR/SM to enable or disable CP capacity for an LPAR, without human intervention.

M

migration rate. The rate (pages/second) of pages being moved from expanded storage through central storage to auxiliary storage.

mintime. The smallest unit of sampling in Monitor III. Specifies a time interval during which the system is sampled. The data gatherer combines all samples gathered into a set of samples. The set of samples can be summarized and reported by the reporter.

mode. Monitor III can run in various modes: GO mode (see *GO mode*) and STOP mode, which is the default mode. See also *graphic mode* and *tabular mode*.

MPL. Multiprogramming level

O

OMVS. Reference to z/OS UNIX System Services

P

partitioned data set (PDS). A data set in direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data.

PDS. partitioned data set

performance management. (1) The activity which monitors and allocates data processing resources to applications according to goals defined in a service level agreement or other objectives. (2) The discipline that encompasses collection of performance data and tuning of resources.

PR/SM. Processor Resource/Systems Manager

Processor Resource/Systems Manager (PR/SM). The feature that allows the processor to run several operating systems environments simultaneously and provides logical partitioning capability. See also *LPAR*.

R

range. The time interval you choose for your report.

Resident time. The time the address space was swapped in, in units of seconds.

RMF monitoring provider. see CIM provider

S

sample. Once in every cycle, the number of jobs waiting for a resource, and what job is using the resource at that moment, are gathered for all resources of a system by Monitor III. These numbers constitute one sample.

SCP. System control program

seek. The DASD arm movement to a cylinder. A seek can range from the minimum to the maximum seek time of a device. In addition, some I/O operations involve multiple imbedded seeks where the total seek time can be more than the maximum device seek time.

service class. In Workload Manager, a subdivision of a *workload*. Performance goals and capacity boundaries are assigned to service classes.

service level agreement (SLA). A written agreement of the information systems (I/S) service to be provided to the users of a computing installation.

Service Level Reporter (SLR). An IBM licensed program that provides the user with a coordinated set of tools and techniques and consistent information to help manage the data processing installation. For example, SLR extracts information from SMF, IMS, and CICS logs, formats selected information into tabular or graphic reports, and gives assistance in maintaining database tables.

service rate. In the system resources manager, a measure of the rate at which system resources (services) are provided to individual jobs. It is used by the installation to specify performance objectives, and used by the workload manager to track the progress of individual jobs. Service is a linear combination of processing unit, I/O, and main storage measures that can be adjusted by the installation.

shared page groups. An address space can decide to share its storage with other address spaces using a function of RSM. As soon as other address spaces use these storage areas, they can no longer be tied to only one address space. These storage areas then reside as *shared page groups* in the system. The pages of shared page groups can reside in central, expanded, or auxiliary storage.

SLA. service level agreement

SLIP. serviceability level indication processing

SLR. Service Level Reporter

SMF. System management facility

SMF buffer. A wrap-around buffer area in storage, to which RMF data gatherers write performance data, and from which the Postprocessor extracts data for reports.

speed. See *workflow*

SRB. Service request block

SRM. System resource manager

SSCH. Start subchannel

system control program (SCP). Programming that is fundamental to the operation of the system. SCPs include MVS, VM, and VSE operating systems and any other programming that is used to operate and maintain the system. Synonymous with *operating system*.

sysplex. A complex consisting of a number of coupled MVS systems.

T

tabular mode. In RMF, the mode in which Monitor III displays performance data in the form of lists. Contrast with *graphic mode*.

TCB. Task control block

threshold. The exception criteria defined on the report options screen.

throughput. A measure of the amount of work performed by a computer system over a period of time, for example, number of jobs per day.

TPNS. Teleprocessing network simulator

TSO. Time Sharing Option, see *Time Sharing Option/Extensions*

Time Sharing Option Extensions (TSO/E). In MVS, a time-sharing system accessed from a terminal that allows user access to MVS system services and interactive facilities.

U

UIC. Unreferenced interval count

uncaptured time. CPU time not allocated to a specific address space.

using. Jobs getting service from hardware resources (PROC or DEV) are *using* these resources.

V

velocity. A measure of how fast work should run when ready, without being delayed for processor or storage access. See also *execution velocity*.

VTOC. Volume table of contents

W

workflow. (1) The workflow of an address space represents how a job uses system resources and the speed at which the job moves through the system in relation to the maximum average speed at which the job could move through the system. (2) The workflow of resources indicates how efficiently users are being served.

workload. A logical group of work to be tracked, managed, and reported as a unit. Also, a logical group of service classes.

WLM. Workload Manager

X

XCF. Cross-system coupling facility

XES/CF. See *Cross-system Extended Services/Coupling Facility*.

Z

zAAP. see IBM System z9 Application Assist Processor.

zIIP. see IBM System z9 Integrated Information Processor.

Notices

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

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

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

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

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

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

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Site Counsel
IBM Corporation
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

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.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Policy for unsupported hardware

Various z/OS elements, such as DFSMS, HCD, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Programming Interface Information

This book documents intended Programming Interfaces that help customers to write their own RMF exit routines and to call RMF functions from their own applications.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at Copyright and trademark information at `www.ibm.com/legal/copytrade.shtml`

Java is a trademark of Sun Microsystems, Inc. in the United States, other countries, or both.

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

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

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

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

Index

Special characters

A

B

C

D

E

F

G

H

I

Q

R

S

T

U

V

W

X

Z

Product Number: 5694–A01

Printed in USA

SC33-7994-14

Spine information:

z/OS

z/OS V1R13.0 RMF Programmer's Guide

Version 1
Release 13

SC33-7994-14