

ABCs of z/OS System Programming Volume 11

Capacity planning

Performance management

RMF, SMF



Paul Rogers
Alvaro Salla

Redbooks



International Technical Support Organization

ABCs of z/OS System Programming Volume 11

December 2010

Archived

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

Archived

Second Edition (December 2010)

This edition applies to Version 1 Release 11 of z/OS (5694-A01) and to subsequent releases and modifications until otherwise indicated in new editions.

© Copyright International Business Machines Corporation 2010. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	vii
Trademarks	viii
Preface	ix
The team who wrote this book	ix
Now you can become a published author, too!	x
Comments welcome	x
Stay connected to IBM Redbooks	x
Chapter 1. Capacity planning overview	1
1.1 Capacity planning definition	2
1.2 Capacity planning introduction	4
1.3 Common mistakes in capacity planning	6
1.4 Continuous availability and capacity planning	8
1.5 Balancing resources for capacity planning	9
1.6 Traditional steps in capacity planning	11
1.7 Capacity planning methods	13
1.8 Projections	15
1.9 Analytic methods	16
1.10 Discrete methods	17
1.11 IBM Large Systems Performance Reference (LSPR)	18
1.12 Internal throughput rate (ITR)	19
1.13 LSPR workloads	22
1.14 Benchmark methods	24
1.15 IBM capacity planning tools	26
1.16 Resource Management Facility overview	29
1.17 zCP3000 introduction	30
1.18 zCP3000 functions	32
1.19 Balanced systems analysis	34
1.20 Capacity planning tools	35
1.21 zCP3000 summary	37
1.22 IBM Processor Capacity Reference for zSeries (zPCR)	38
1.23 Getting started with zPCR	40
1.24 zPCR function selection	42
1.25 zPCR Reference-CPU	43
1.26 Choosing a workload mix	44
1.27 zPCR choosing a workload mix Help panel	45
1.28 LPAR Host and Partition Configuration panel	47
1.29 LPAR configuration capacity planning	49
1.30 zPCR graphics	51
Chapter 2. Performance management	53
2.1 Service level agreement (SLA)	54
2.2 Performance analysis overview	55
2.3 Performance management heuristics	57
2.4 What to measure	59
2.5 General performance management metrics	61
2.6 Average transaction response time	63
2.7 Transaction response time components	65

2.8	Using and Delay sampling information	68
2.9	External throughput rate (ETR)	69
2.10	Resource utilization	71
2.11	Physical PU utilization example	73
2.12	Channel utilization example	74
2.13	Saturation design point (SDP)	75
2.14	Processor performance metrics	77
2.15	CPU time	78
2.16	z10 Cycle Time and Server Time Protocol	80
2.17	Non-CPU time-related metrics	82
2.18	Millions of instructions per second (MIPS)	83
2.19	CPU service units	85
2.20	RMF Partition Data report	88
2.21	Internal throughput rate (ITR)	90
2.22	Large Systems Performance Reference (LSPR)	92
2.23	LSPR relative processor power (RPP)	94
2.24	CP-measurement facility highlights	96
2.25	Hardware and software requirements for CPMF	98
2.26	CPMF counters and sampling	100
2.27	Set of instructions to interface with CPMF	102
2.28	CPU-measurement counter facility	103
2.29	How CPMF works	106
2.30	Setting up hardware data collection	107
2.31	PU caching z10 highlights	109
2.32	z10 cache performance aspects	111
2.33	Cache page states	113
2.34	Using CPMF caching counters data	114
2.35	Virtual storage concept	116
2.36	Program status word (PSW) format	118
2.37	Dynamic address translation (DAT)	121
2.38	Translating a 31-bit virtual address	123
2.39	Translating a 64-bit virtual address	124
2.40	Large page support	126
2.41	Translation lookaside buffer (TLB)	127
2.42	TLB performance and 1 MB pages	129
2.43	Large page support specifications	131
2.44	CPMF extended counters on TLB	133
2.45	IBM common cryptographic architecture (CCA)	134
2.46	Symmetric cryptography	136
2.47	Asymmetric cryptography	138
2.48	Clear key and non-clear key algorithms	139
2.49	Hardware crypto in a z10	140
2.50	z10 multichip module (MCM)	142
2.51	z10 PU chip	143
2.52	CPAF crypto algorithms	144
2.53	CPMF crypto counters	146
2.54	Crypto counters example	148
2.55	CPU-measurement sampling facility	149
2.56	CPU-measurement sampling facility functions	150
2.57	CPMF sample data block tables	152
2.58	CPMF sample data block (SDB)	153
2.59	Sample data block data entry	154
2.60	I/O performance and metrics SLA	156

2.61	z/OS CPU time accounting	159
2.62	z/OS preemptability	161
2.63	Formulas and laws in performance management	163
2.64	Little's Law	164
2.65	Markov's Equation	166
2.66	DASD saturation point	168
2.67	Pareto's 80/20 Rule	169
2.68	80/20 rule and service class periods	170
2.69	Partition's Law	171
2.70	z10 EC SRM Constants	173
2.71	Principle of Locality	174
Chapter 3. Resource Measurement Facility		175
3.1	Resource Measurement Facility overview	176
3.2	RMF Performance Management panel	178
3.3	RMF monitors	179
3.4	RMF Monitor I	181
3.5	RMF Monitor II	182
3.6	RMF Monitor II ARD report	183
3.7	RMF Monitor III primary panel	184
3.8	RMF Monitor III contention analysis	186
3.9	RMF Workflow Exceptions report	188
3.10	Definition and Criteria panel	190
3.11	RMF Postprocessor	192
3.12	RMF Spreadsheet Reporter	194
3.13	RMF Spreadsheet Reporter panels	196
3.14	RMF Performance Monitor	197
3.15	RMF Distributed Data Server (DDS)	199
3.16	RMF Monitor III Data Portal for z/OS	201
3.17	RMF Monitor III Data Portal for z/OS Overview panel	203
3.18	RMF Monitor III Data Portal for z/OS Explore panel	204
3.19	RMF Monitor III Data Portal for z/OS Metrics Explore panel	205
3.20	RMF Portal CACHDET report panel	207
3.21	RMF Monitor III Data Portal for z/OS My View panel	209
3.22	RMF enhancements for z/OS V1R11	210
3.23	Cryptographic hardware information	212
3.24	Cryptographic coprocessors	214
3.25	RMF Workload Activity - Crypto (CRY)	216
3.26	RMF Coupling Facility reports	217
3.27	CF-to-CF activity reporting	219
3.28	HiperSockets: Client/Server in a mainframe machine	220
3.29	RMF enhancement to support HiperSockets	222
3.30	System Management Facility (SMF)	224
3.31	Using SMF data records	225
3.32	Importance of SMF records	227
3.33	Importance of SMF records	229
3.34	Recording SMF data records	231
3.35	SMF recording to DASD (SYS1.MANx)	232
3.36	SMF recording to SYS1.MANx	234
3.37	SMF on System Logger	236
3.38	Where System Logger stores data	238
3.39	Customizing SMF	239
3.40	Which SMF records to record	241

3.41 System Logger log streams	243
3.42 Dumping SMF data sets	245
3.43 Dumping SMF records - log streams	247
3.44 Dumping selective SMF records	249
Related publications	251
IBM Redbooks publications	251
Other publications	251
Online resources	251
How to get IBM Redbooks publications	252
Help from IBM	252

Notices

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

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

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

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

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

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

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

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

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


COPYRIGHT LICENSE:

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

Trademarks

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

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

AIX®	Lotus®	SNAP/SHOT®
C/370™	MVS™	Sysplex Timer®
CICS®	OS/390®	System z10®
DB2®	OS/400®	System z9®
Domino®	Parallel Sysplex®	System z®
DRDA®	PR/SM™	System/390®
DS8000®	Processor Resource/Systems Manager™	Tivoli®
eServer™	pSeries®	VTAM®
FICON®	RACF®	WebSphere®
GDDM®	Redbooks®	z/Architecture®
Geographically Dispersed Parallel Sysplex™	Redpaper™	z/OS®
HiperSockets™	Redbooks (logo)  ®	z/VM®
Hiperspace™	Resource Link™	z/VSE™
IBM®	Resource Measurement Facility™	z10™
IMS™	RMF™	z9®
Language Environment®	S/390®	zSeries®

The following terms are trademarks of other companies:

Adobe, the Adobe logo, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

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

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

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

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

Preface

The ABCs of z/OS® System Programming is a thirteen-volume collection that provides an introduction to the z/OS operating system and the hardware architecture. Whether you are a beginner or an experienced system programmer, the ABCs collection provides the information you need to start your research into z/OS and related subjects. If you want to become more familiar with z/OS in your current environment, or if you are evaluating platforms to consolidate your e-business applications, the ABCs collection will serve as a powerful technical tool.

The volumes contain the following content:

Volume 1: Introduction to z/OS and storage concepts, TSO/E, ISPF, JCL, SDSF, and z/OS delivery and installation

Volume 2: z/OS implementation and daily maintenance, defining subsystems, JES2 and JES3, LPA, LNKLST, authorized libraries, Language Environment®, and SMP/E

Volume 3: Introduction to DFSMS, data set basics, storage management hardware and software, VSAM, System-Managed Storage, catalogs, and DFSMSStvs

Volume 4: Communication Server, TCP/IP and VTAM®

Volume 5: Base and Parallel Sysplex®, System Logger, Resource Recovery Services (RRS), global resource serialization (GRS), z/OS system operations, automatic restart management (ARM), Geographically Dispersed Parallel Sysplex™ (GPDS), availability in the zSeries® environment

Volume 6: Introduction to security, RACF®, Digital certificates and PKI, Kerberos, cryptography and z990 integrated cryptography, zSeries firewall technologies, LDAP, Enterprise Identity Mapping (EIM), and firewall technologies

Volume 7: Printing in a z/OS environment, Infoprint Server and Infoprint Central

Volume 8: An introduction to z/OS problem diagnosis

Volume 9: z/OS UNIX® System Services

Volume 10: Introduction to z/Architecture®, zSeries processor design, zSeries connectivity, LPAR concepts, HCD, and HMC

Volume 11: Capacity planning, performance management, RMF™, and SMF

Volume 12: WLM

Volume 13: JES3

The team who wrote this book

This IBM® Redbooks® publication was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Paul Rogers is a Consulting IT Specialist at the International Technical Support Organization, Poughkeepsie Center and has worked for IBM for 42 1/2 years. He writes extensively and teaches IBM classes worldwide on various aspects of z/OS JES3, zFS, Infoprint Server, and z/OS UNIX. Before joining the ITSO 22 years ago, Paul worked in the IBM Installation Support Center (ISC) in Greenford, England, providing OS/390® and JES support for IBM EMEA and the Washington Systems Center in Gaithersburg, Maryland.

Alvaro Salla is an IBM retiree. He worked for IBM for more than 34 years, focusing on large systems. He has co-authored numerous Redbooks and spent many years teaching about large systems, from S/360 to S/390®. Alvaro holds a Chemical Engineering degree from the University of Sao Paulo, Brazil.

Note: This edition, SG24-6327-01, was updated by Paul Rogers and Alvaro Salla.

Now you can become a published author, too!

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

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

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

ibm.com/redbooks

- ▶ Send your comments in an email to:

redbook@us.ibm.com

- ▶ Mail your comments to:

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

Stay connected to IBM Redbooks

- ▶ Find us on Facebook:

<http://www.facebook.com/pages/IBM-Redbooks/178023492563?ref=ts>

- ▶ Follow us on Twitter:
<http://twitter.com/ibmredbooks>
- ▶ Look for us on LinkedIn:
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:
<http://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS feeds:
<http://www.redbooks.ibm.com/rss.html>

Archived

Archived



Capacity planning overview

This chapter discusses the concept of capacity planning and how it relates to workload processing. It also provides an overview of the mechanisms used to perform effective capacity planning.

The chapter covers the following subjects:

- ▶ Capacity planning introduction
- ▶ Balancing resources for capacity planning
- ▶ Capacity planning methods
 - Guidelines
 - Projection method
 - Analytic method
 - Discrete method
 - Benchmarks
- ▶ IBM capacity planning tools used by our technical support team
- ▶ RMF as a capacity planning information provider and RMF integrations
- ▶ zCP3000
- ▶ LSPR concepts

1.1 Capacity planning definition

- ❑ Capacity planning ensures that adequate resources are available in the future, for critical workload to complete in an appropriate time
- ❑ Following are comments about capacity planning:
 - Performance analysis is short term (3-7 Days)
 - Capacity planning is long term (6-24 months)
 - A service level agreement is a necessity
 - Transactions are ordered by priority or importance
 - Discretionary workloads

Figure 1-1 Capacity planning definition

Capacity planning definition

Capacity planning is a discipline that ensures adequate resources are available in the future for the critical workload to complete in an appropriate time. For capacity planning, you try to predict how changes in workload will change the requirements for all resources.

Such planning might be undertaken to predict future requirements as additional workload is brought into the system. Alternatively, it might be undertaken to predict the impact to signaling resources if the workload is redistributed around the sysplex in a different configuration.

By “resources” we mean processors, main storage, Coupling Facilities, Coupling Facility links, channels, I/O controllers, DASD space, DASD I/O rate, tape, printers, and the network.

Performance analysis

Performance analysis focuses on serving critical workload during a short-term period (for example, 3 to 7 days). Capacity planning focuses on serving critical workload in the long term (6 to 24 months).

Service level agreement

A service level agreement (SLA) is a contract that objectively describes measurables such as:

- ▶ Average transaction response time for network, I/O, CPU, or total
- ▶ The distribution of these response times (for example, 90% TSO trivial at less than 0.2 of a second)

- ▶ Transaction volumes
- ▶ System availability

A *transaction* is a business unit of work, for example a CICS® user interaction or a batch job. Ideally, a transaction is defined from a user's point of view.

The definition and implementation of an SLA can be done in an installation in a more or less formal way, but more precisely by:

- ▶ The expectations of the users.
- ▶ The capabilities of the computer shop; when these have been defined, then tracking and monitoring are easier.

This definition is important with regard to the capabilities of performance management in a z/OS system. There, Workload Manager enables you to specify explicit performance goals for your applications, and the reporting capabilities within RMF will allow you to track them directly.

Performance management means monitoring and allocating data processing resources to applications according to service level agreements or informal objectives. It involves an ongoing cycle of measuring, planning, and modifying. Workload management provides z/OS performance management externals in a service policy that reflects goals for work, expressed in terms commonly used in service level agreements. Because the terms are similar to those commonly used in an SLA, you can communicate with users, with business partners, and with z/OS, using the same terminology.

Service level objective

A service level objective (SLO) is a key element of a service level agreement (SLA) between a service provider and a client. SLOs are agreed as a means of measuring the performance of the service provider and are outlined as a way of avoiding disputes between the two parties based on misunderstanding.

There is often confusion in the use of SLA and SLO.

- ▶ Service level agreements refer to the entire agreement that specifies what service is to be provided, how it is supported, times, locations, costs, performance, and responsibilities of the parties involved.
- ▶ Service level objectives are specific measurable characteristics of the SLA such as availability, throughput, frequency, response time, or quality.

The service level objective specification has two components: the promised *response* or turnaround time of a unit of work, and the maximum *rate* for the unit of work. This is because it cannot be promised that a transaction will complete in, for example, less than 1 second, without putting a limit on the number of transactions. No collection of resources can promise such a threshold for an unlimited number of transactions. Therefore, both responsiveness and rate are specified.

Refer to 2.1, “Service level agreement (SLA)” on page 54 for more information about these related disciplines.

1.2 Capacity planning introduction

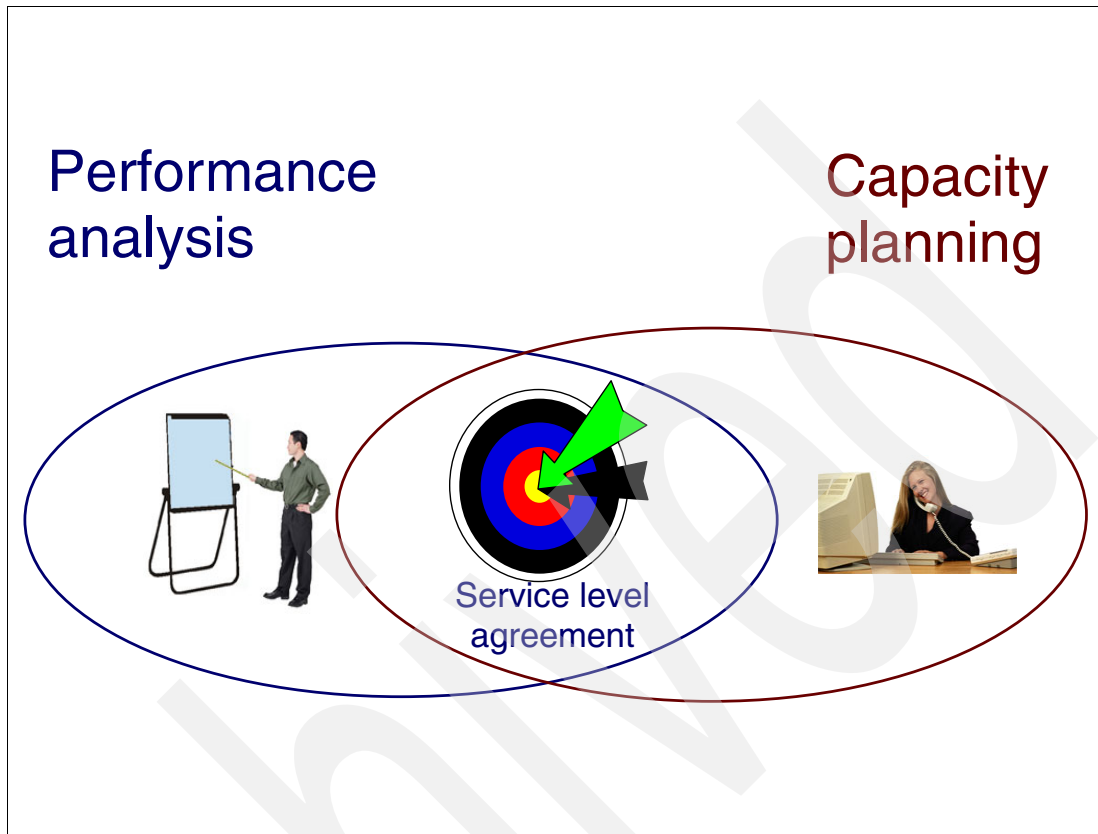


Figure 1-2 Capacity planning concepts

Capacity planning concepts

Planning for capacity involves estimating the resources needed for future workloads. The mission of *capacity planners* is to ensure that there are adequate resources available to meet current and future workload demands. They must forecast to determine how much additional capacity will be needed in the future based on increasing workload demands. Along with performance analysis (Figure 1-2), capacity planning depends on the statements declared in a SLA. The SLA is the threshold which gives the data processing user permission to complain if performance falls below that threshold. 2.2, “Performance analysis overview” on page 55 for further information about performance analysis.

Capacity planning has been an essential task since commercial computing began. It has always been imperative for companies to plan the capacity needed to adequately process workloads that provide service to users. Large IT installations have a group in charge of capacity planning, and system programmers performing this activity are known as capacity planners.

Capacity planners

Generally, capacity planners are analysts who have extensive system experience. However, there are several subactivities within capacity planning tasks where staff with less system experience but who have other skills can effectively contribute. For example, it is common to use mathematical models (which you may already be familiar with) to predict resource consumption. It is also common to use products from non-mainframe platforms to collect, analyze, and report statistics in a format that other departments can understand.

Many capacity planners develop their own tools to help them predict and demonstrate the needs of an IT department.

System programmers and support analysts

The systems programmers (or support analysts) in charge of performance analysis work very closely with capacity planners. These analysts understand the same statistical language and use almost the same sources of information to create unique databases. To plan effectively, capacity planners have to know the size of new applications and the periodic growth of all applications and subsystems. They need to be able to predict not only future CPU size (to process applications), but also the I/O rate, net traffic, and peripherals.

Effective capacity planning also involves being aware of future IT plans at the enterprise level. Capacity planners also may provide advice about consolidation and distribution of processing data, server clustering, and the best platform for applications.

More about service level agreements

Performance administration is the process of defining and adjusting performance goals. Workload management introduces the role of the service level administrator. The service level administrator is responsible for defining the installation's performance goals based on business needs and current performance. This explicit definition of workloads and performance goals is called a *service definition*. Installations can already have this kind of information in a service level agreement (SLA). The service definition applies to all types of work, including CICS, IMS™, TSO/E, z/OS UNIX System Services, JES, APPC/MVS™, LSFM, DDF, DB2®, SOM, Internet Connection Server (also referred to as IWEB) and others. You can specify goals for all MVS-managed work, whether online transactions or batch jobs. The goals defined in the service definition apply to all work in the sysplex.

Workload management provides new MVS performance management externals in a service policy that reflects goals for work, expressed in terms commonly used in service level agreements. Because the terms are similar to those commonly used in an SLA, you can communicate with users, with business partners, and with MVS using the same terminology.

1.3 Common mistakes in capacity planning

- ❑ Forgetting the concept of balanced systems
- ❑ Ignoring latent demand
 - Latent demand is work that is hidden and waiting to be unleashed
- ❑ Being out of touch with business forecasts
- ❑ Ignoring complexity

Figure 1-3 Common mistakes in capacity planning

Common mistakes in capacity planning

There are several mistakes commonly encountered when performing capacity planning, as explained here.

Forgetting the concept of balanced systems

Keep in mind that it takes a combination of resources to process work: CPU, processor storage, and I/O, so increasing the capacity of one resource by itself may not necessarily allow more work to be processed.

Ignoring latent demand

Latent demand is work that is hidden and waiting to be unleashed. There are two types of latent demand:

- ▶ One type is that which is already in the system and can be evaluated. You can begin quantifying latent demand by noting your peak-to-average ratio (for example, peak hour CPU busy compared to prime-shift average CPU busy). This ratio will drop as latent demand builds, so track it and compare it to your current system. Relieving any bottleneck (for example, by upgrading constrained CPU, increasing network bandwidth, or improving I/O subsystem) will release latent demand.
- ▶ The other type is that which is not already in the system. It is in the actual business, mainly because of long response times for all previous transactions. This type of latent demand is difficult to evaluate.

Being out of touch with business forecasts

This mistake is obvious, but can be difficult to avoid. Capacity planners need to stay informed of business plans that will impact IT resource requirements (for example, departmental growth, new applications, and mergers).

Note: IT growth is not necessarily business growth, although it can be correlated. IT growth can occur simply to maintain market share.

Ignoring complexity

Factors other than transaction growth will increase demand on IT resources. Enhancements to existing applications, new government regulations, growing databases and so on can all result in an increase in resource consumption. This means that the same transaction running today may require 10-15% more resource than it did last year.

1.4 Continuous availability and capacity planning

- CP, IFL, ICF, zAAP and zIIP processors, if there are spare PUs available on any installed book
- The Capacity on Demand (CoD) On/Off plan can be used for the following:
 - Additional PU books
 - LIC - Configuration Control (LIC-CC) provides for server upgrade with new microcode
 - Memory - requires available capacity on installed memory cards
 - I/O cards ports and channels

Figure 1-4 z10™ non-disruptive upgrades

Continuous availability and capacity planning

Capacity planning is now simultaneously both more complex and also easier:

- ▶ The complexity derives from the difficulty of predicting business growth due to an unstable economy and to the potential growth of Internet business (e-business).
- ▶ The ease derives from z10 technology, in which almost all hardware components can receive nondisruptive upgrades. This means that any subdimensioned prediction can be fixed without disrupting the continuous availability (24 x 7) of the systems. No Power-on Resets (PORs), logical partition deactivations, or IPLs are needed. Every hardware upgrade, however, involves financial considerations. The following list shows the hardware components allowing nondisruptive upgrades:
 - CP, IFL, ICF, zAAP and zIIP processors can be concurrently added to a z10 server if there are spare PUs available on any installed book. The Capacity on Demand (CoD) On/Off plan can be used for this task.
 - Model upgrades within the server family are accomplished by installing additional books. Books, being on separate power boundaries, are physically isolated from each other, thereby allowing them to be plugged and unplugged independently.
 - LIC - Configuration Control (LIC-CC) provides for server upgrade with no hardware changes by enabling the activation of additional, previously installed capacity.
 - Memory - this requires available capacity on installed memory cards.
 - I/O cards ports and channels.

1.5 Balancing resources for capacity planning

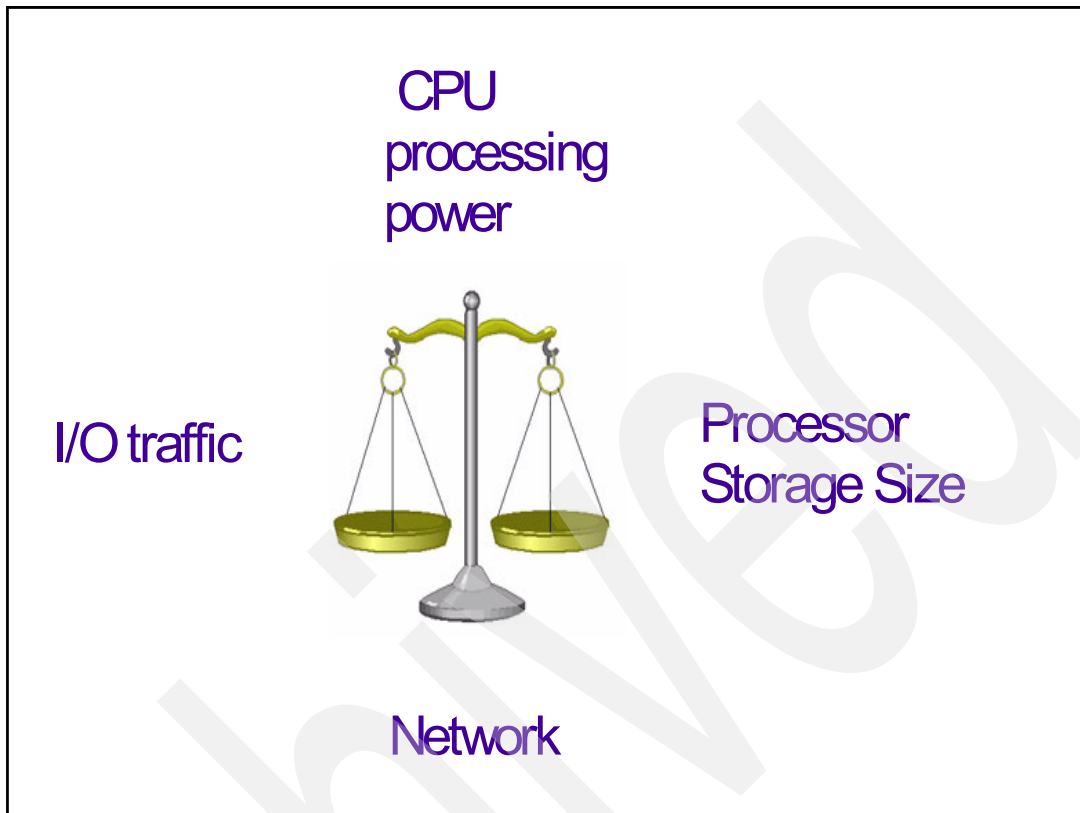


Figure 1-5 Balancing resources for capacity planning

Balancing resources for capacity planning

Capacity planning resources, in a balanced system, are not independent variables in the determination of system performance. Instead, the interrelationship of processing power, I/O capability, processor storage size, and networks determines system performance. Capacity can then be determined based on performance objectives. In z/OS, the workload manager manages the achievement of user objectives.

Capacity planning input

The capacity planning process has two distinct sets of inputs.

- ▶ The technical side includes:
 - Data model showing the hardware and software configuration
 - Data describing the resource utilization (usually SMF records) per sysplex, per logical partition, per CEC. This data populates the data model.

To make the results delivered by a capacity planning task reliable, capture all this data from a balanced system. If you have a concrete I/O bottleneck for example, your CPU consumption figures are not reliable. Having captured performance data in a balanced system avoids the proverbial “garbage in, garbage out” issue.

- ▶ Business side includes:
 - Business forecast about growth
 - Cost model

Computing resources are not independent variables. Instead, the interrelationship of processing power, I/O capability, processor storage size, and networks determines system performance.

Balanced systems

A *balanced system* is one in which all resources are dimensioned toward each other, thereby optimizing the system's capacity and achieving the best performance and throughput in accordance with the goals. A balanced system should have no single bottleneck. However, if a single bottleneck is unavoidable, then it should be the most expensive one to eliminate (for example, a CPU-constrained system); all other bottlenecks are to be eliminated.

Balanced systems provide these benefits:

- ▶ They optimize your enterprise's IT investments.
- ▶ They increase your IT productivity.
- ▶ They increase communication between users and WLM services.
- ▶ They provide more transparency for management by establishing service level agreements (SLAs).
- ▶ They help to avoid performance problems that can seriously impact your business.

1.6 Traditional steps in capacity planning

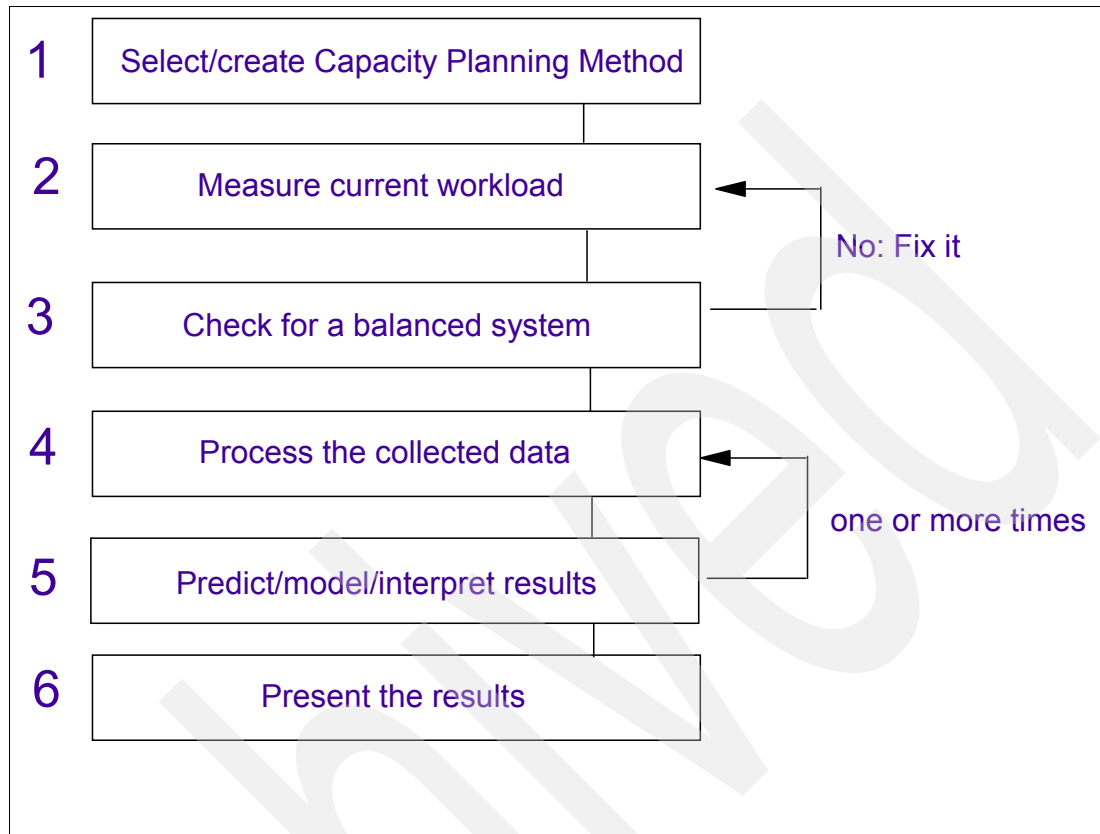


Figure 1-6 Traditional steps in capacity planning

Steps in capacity planning

There are several methods and tools that you can use when performing capacity planning. Regardless of the method used, the steps illustrated in Figure 1-6 are generally required.

Select or create a capacity planning method

There are several methods available to accomplish the capacity planning task; see 1.7, “Capacity planning methods” on page 13 for additional information about this topic. The methods most commonly used by specialists are the analytic model and linear projections.

Measure current workload

Measuring the current workload means that you capture data about the behavior of the workloads in the running system. The most commonly used type of data is the SMF records maintained in a database, which will be used as input by the capacity planning tool. To process this collected data, clients can use worksheets or tools that are available in the market. Clients are increasingly relying on the IBM zCP3000 tool, where the processing step is executed by IBM personnel together with a client counterpart.

Check for a balanced workload

It is always necessary to check the health of the system being measured to avoid undesired deviations. Refer to 1.5, “Balancing resources for capacity planning” on page 9 for more information about this topic.

Process the collected data

Usually this step refers to processing the huge amount of performance data kept in VSAM files or database table spaces. The first task is to perform data extraction, in which valuable data covering the key intervals is isolated. Using the software tools described in 1.15, “IBM capacity planning tools” on page 26 produces graphics and tables to be analyzed.

Predict/model/interpret results

This step involves using the graphic and tables produced by the previous step, and calibrating the model you have defined. The results are to be interpreted and translated to real hardware resources. If this step does not result in meeting expectations, you must process the collected data again.

Present the results

Discussions of capacity planning often emphasize the final step, which is presenting your results. Keep in mind that high-level executives are generally more interested in an overview than in the planning details. Using clear graphics and showing only relevant statistics can help communicate your results most effectively.

1.7 Capacity planning methods

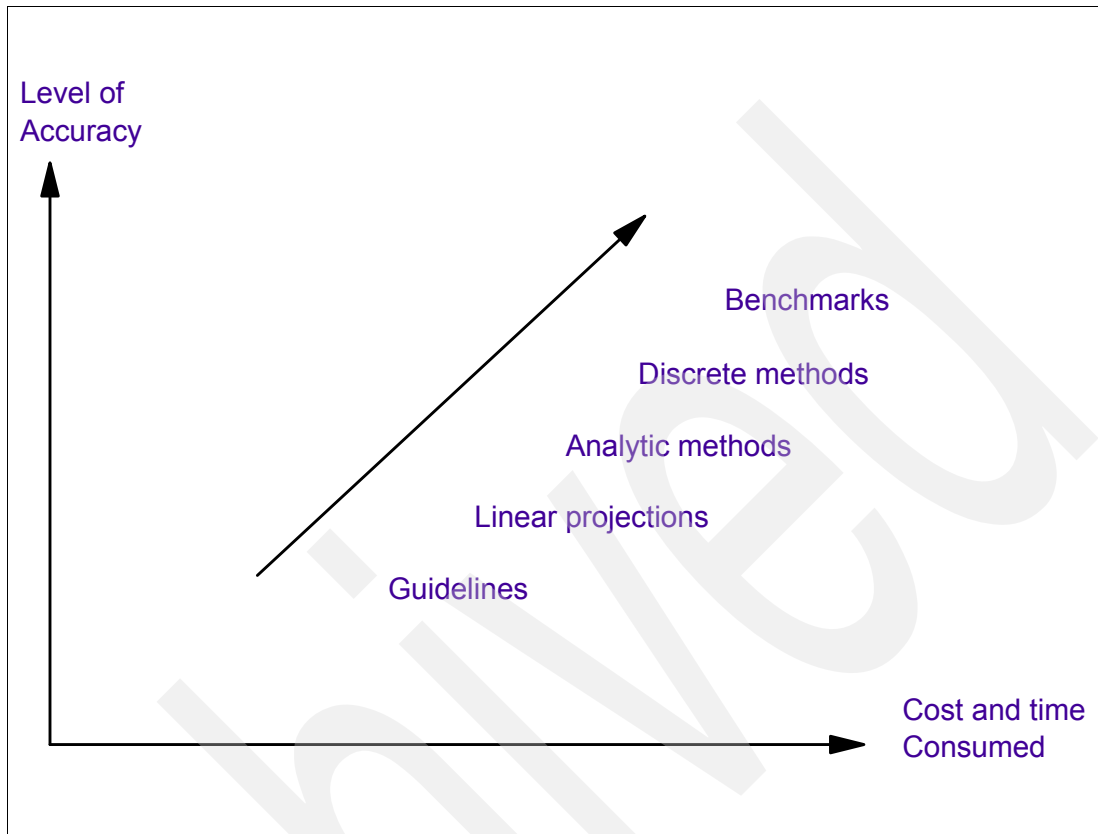


Figure 1-7 Capacity planning methods

Capacity planning methods

As mentioned, several methods can be used for capacity planning. Depending on the time, cost, skill, and level of detail and accuracy that is available, capacity planners choose the method that best meets their expectations. This section lists and describes those methods to help you determine which is most suitable to your situation.

Guidelines

Guidelines range somewhere between a guess and a rule of thumb and they are simple to execute. Simple, informal projections are often made without documentation and may be based on experience and knowledge of the system.

Linear projections

Linear projections range from simple diagrams created with spreadsheets to more sophisticated methods.

Analytic methods

Based on mathematical methods such as the queuing theory, analytic methods can provide insight into forecasting CPU utilization, response time evaluation, capture ratios, effect of buffering, effect of queueing, and so forth. Based on measured data from today's environment, various configurations and growth scenarios are studied and documented.

Discrete methods

Discrete methods are applications of discrete simulation. Unlike analytic simulation, discrete simulation is not based on mathematical formulas. Using discrete capacity planning methods, dissimilar workloads and their effect on each other are modeled. An example is the effect of combining batch and interactive workloads on the same pSeries® system. Sometimes a discrete simulation requires more detailed input data than an analytic simulation.

Note: Analytic and simulation tools expect a 30% accuracy for response time.

Benchmarks

Different modeling approaches are used to achieve different goals. The best modeling solution must be matched to the objectives trying to be solved. One of the most common approaches is a benchmark. With a benchmark, you can achieve accurate performance prediction results by building a real environment with the expected number of users processing real applications using real data. However, the financial and time commitment required for a true benchmark often outweighs the benefit of the results.

1.8 Projections

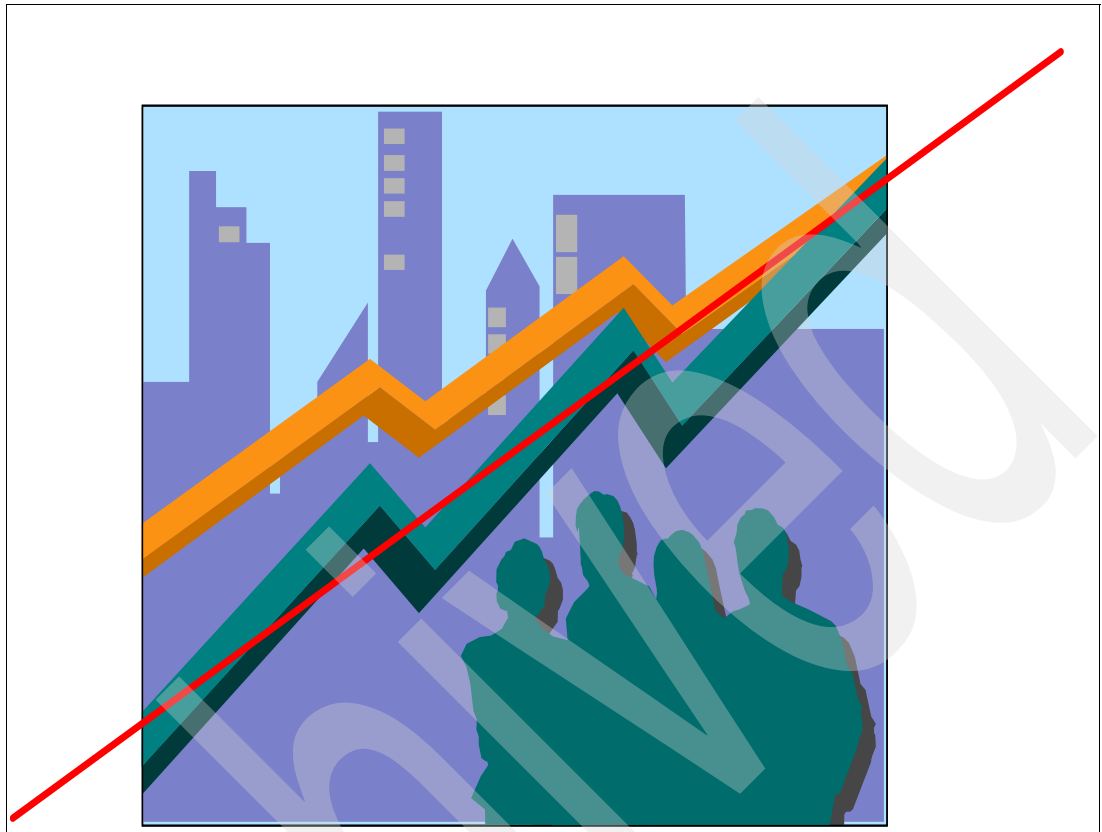


Figure 1-8 The projection method

Using projections for capacity planning

A linear projection has better accuracy than a guideline, but it does not apply for all cases. Often there is no linear correlation between the variables involved in capacity planning measurements. For example, newer types of workloads such as WebSphere applications or the unpredictable peaks of transactions can easily transform a forecast into an unreliable assumption. A useful application of linear projection is when historical workload utilization is plotted and a straight line is feasible to predict future workload utilization

Capacity planning tools

The following tools (based on Large Systems Performance Reference (LSPR) data) are examples of tools that can help you estimate various configurations and workloads using a spreadsheet-like approach:

- ▶ LPAR Capacity Estimator (LPAR/CE)
- ▶ zCP3000 Quick Sizer
- ▶ Processor Capacity Reference (PCR)
- ▶ zProcessor Capacity Reference

These tools are based on previous IBM comparisons among models of mainframes, resulting in an Internal Throughput Ratio (ITR) for each machine and an Internal Throughput Rate Ratio (ITRR) when comparing to a base model in terms of performance capacity.

1.9 Analytic methods

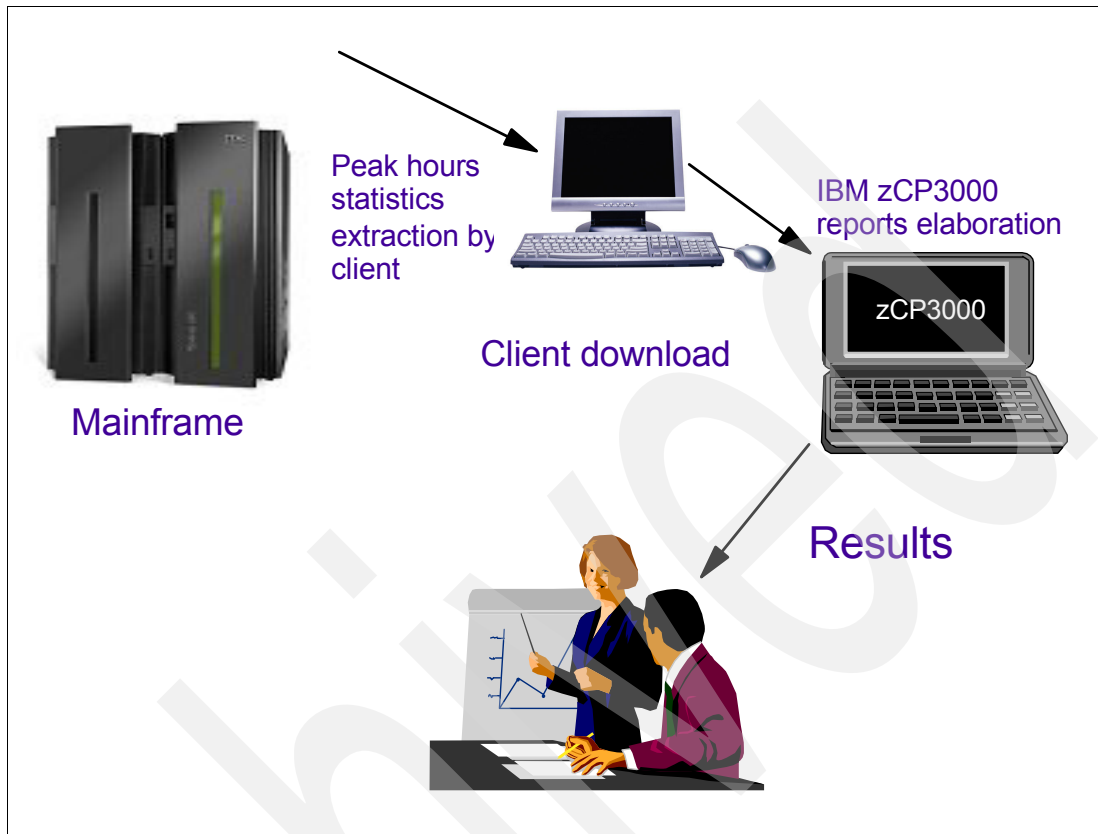


Figure 1-9 Analytic methods

Using analytic methods for capacity planning

Analytic methods are based on mathematical models such as queueing theory and statistics formulas (such as Erlang). Analytic methods can provide insight into forecasting CPU utilization, response time evaluation, capture ratios, effect of buffering, effect of queueing, and so forth. Based on measured data from today's environment, various configurations and growth scenarios are studied and documented. Generally, analytic tools have an accuracy bandwidth between 8% to 15% for utilization, depending on the scenario, level of detail, and provided data.

zCP3000, Tivoli Performance Modeler for z/OS and BEST/1 are a subset of the many IBM and other tools that fall into the analytic methods category. zCP3000 is an internal IBM Tool that analyses data from the client and generates reports that show a kind of "snapshot" of main resource utilization.

Using zCP3000

Using zCP3000, the client decides which peak workload to analyze and then extracts data from the System Management Facility (SMF). IBM produces reports, based on that collected data, which show normal peak utilization of main resources such as CPU, I/O and Storage. This allows the evaluation of future growth areas and the visualization of where new capacity might be needed. zCP3000 uses Large Systems Performance Reference (LSPR) data for machine capacity positioning purposes. zCP3000 and LSPR are described in more detail in 1.15, "IBM capacity planning tools" on page 26.

1.10 Discrete methods

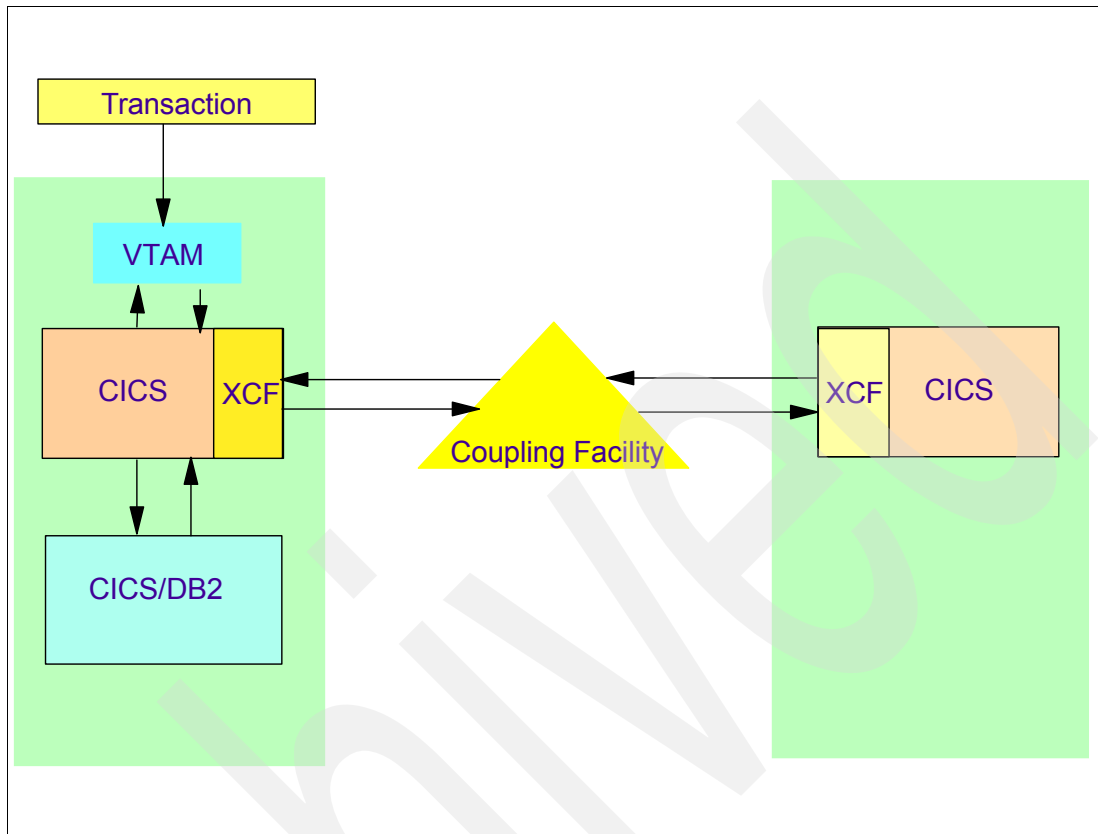


Figure 1-10 Transaction life events for the discrete simulation method

Using the discrete method for capacity planning

A *discrete event simulation* employs a next-event technique to control the behavior of the model. Many applications of discrete simulation involve queuing systems of one kind or another. The queuing structure may be obvious, as in a queue of jobs waiting to be processed on a batch computer or in a stack of aircraft waiting for landing space at an airport. Other examples are clients waiting in line for service at a bank or grocery store. In the simplest case, the queue operates with a first-in, first-out (FIFO) discipline.

In capacity planning, a discrete method can be used as an application of the discrete simulation. Unlike analytic simulation, discrete simulation is not based on mathematical formulas. Using discrete capacity planning methods, dissimilar workloads and their effect on each other are modelled. An example is the effect of combining batch and interactive workloads on the same z/OS system. A discrete simulator simulates events that take place in a system, such as the input of a message to the system, its arrival at the CPU, the instructions required to process it, polling the terminal, and sending the response. All these events are simulated, tabulated, and reported by the discrete simulation model.

SNAP/SHOT simulation tool

Sometimes a discrete simulation requires more detailed input data than an analytic simulation. SNAP/SHOT is an example of a simulation tool. Discrete simulation tools usually have an accuracy bandwidth of 5% to 10% for utilization. However, the accuracy may be +/- 30% for response time. This depends on the scenario complexity and level of detail.

1.11 IBM Large Systems Performance Reference (LSPR)

- ❑ Large Systems Performance Reference (LSPR)
 - An IBM project using the benchmark methodology to estimate the PU capacity needed for your workload
- ❑ Measures internal throughput rates (ITR) for:
 - Different machines with different workload types
 - Different operating systems
 - Measurements are made with z/OS, z/VM, z/VSE
- ❑ Based on the idea of comparing the same set of workloads on several CECs

Figure 1-11 Large Systems Performance Reference (LSPR)

IBM Large Systems Performance Reference (LSPR) method

The IBM Large Systems Performance Reference (LSPR) method is designed to provide relative processor capacity data for IBM System/370™, System/390™, and z/Architecture™ processors, both IBM and IBM-compatible. All LSPR data is based on a set of measured benchmarks and analysis, covering a variety of system control program (SCP) and workload environments. LSPR data is intended to be used to estimate the capacity expectation for a production workload when considering a move to a new processor. IBM considers LSPR data to be a reliable and comprehensive set of relative processor capacity data. This is the only reference of its type that is based primarily on actual processor measurements. Because it is based on measurements, LSPR data takes into account individual SCP and workload sensitivities to the underlying design of each processor represented.

The LSPR ratios represent an IBM assessment of relative processor capacity in an unconstrained environment for the specific benchmark workloads and system control programs specified in the tables. Ratios are based on measurements and analysis. The amount of analysis as compared to measurement varies with each processor. The data that follows is based solely on IBM measurements and analysis of the processors in the tables. For machines not listed in the tables, contact the corresponding vendor for performance information.

1.12 Internal throughput rate (ITR)

- ❑ ITR is a CPU metric used in the LSPR project and it determines the capacity of a processor in terms of the number of executed transactions
 - $ITR = \# \text{ of ended transactions} / \text{CPU_Time consumed}$
- ❑ ITR is a function of:
 - CPU speed - the faster the CPU, the higher ITR
 - The more efficient the operating system is, the higher the produced ITR
 - Short trivial transactions produce higher ITRs - TSO transactions produce a different ITR from CICS transactions
- ❑ One derivation of ITR is:
 - Number of ended transactions/MIPS consumed

Figure 1-12 Internal throughput rate

Internal throughput rate

An estimated internal throughput rate (ITR) is considered to have an accuracy level that is very close to that of actual measurements. Estimates are not provided in the LSPR without a high degree of confidence in the process, and in the resulting ITR values.

The major challenge of measuring CPU speed is that for any CEC, the speed depends heavily on the set of executed instructions. CPU time consumed by one set of key business transaction is the best metric to measure the speed of a CPU.

However, it is impractical to use because CPU time consumed is not repetitive when the same program runs in different processors. To address this issue, IBM introduced the ITR metric. It is defined through the following formula:

$$ITR = \text{Number of ended Transactions} / \text{CPU_Time consumed}$$

ITR is more suitable for being a CPU metric than ETR because ITR is less sensitive to a much larger number of variables. The set of variables that affect ITR are depicted in Figure 1-12.

Thus, if you keep constant the same operating system (z/OS V1R11, for example) and vary the transaction workload, you can use ITR to measure the CPU speed of a processor in terms of transactions. IBM calculates ITRs for several workloads and machines using the Large Systems Performance Reference (LSPR) methodology.

ITR numbers are measurements, by workload type, which determine the capability of a machine in terms of the number of transactions per CPU second. This is important because processor capability varies according to the workload mix.

$$\text{ITR} = \frac{\text{Number of transactions}}{\text{CPU time}}$$

External Throughput Rate

Any bottleneck, whether internal or external to the system, will affect the ETR. Examples include I/O constraints, tape mount delay, and paging delay. Therefore, it is more difficult to obtain a repeatable measure than with ITR. IBM calculates ITRs for several workloads and machines using the Large Systems Performance Reference (LSPR) methodology. ITR provides a reliable basis for measuring capacity and performance. The relationship between ETR and ITR is:

$$\text{ETR} = \text{ITR} \times \text{CPU Utilization}$$

When the CPU utilization is 100%, the ETR is equal to the ITR. Using ITRs, you can define an ITR ratio (ITRR) between two CEC processors.

IBM Large Systems Performance Reference ratios

IBM Large Systems Performance Reference (LSPR) ratios represent an IBM assessment of relative processor capacity (through ITRs) in an unconstrained (no bottlenecks) environment for the specific benchmark workloads and system control programs specified in the tables. Ratios are based on measurements and analysis. The amount of analysis as compared to measurement varies with each processor.

Many factors, including but not limited to the following, may result in variances between the ratios provided by LSPR and actual operating environments:

- ▶ Differences between the specified workload characteristics and your operating environment
- ▶ Differences between the specified system control program and your actual system control program
- ▶ Incorrect assumptions in the analysis
- ▶ Unknown hardware defects in processors used for measurement
- ▶ Different number of z10 books for the same number of active PUs

IBM does not guarantee that your results will correspond to the ratios provided at the LSPR. That information is provided “as is” and without warranty, express or implied. LSPR data for IBM processors is generally made available at announce time. IBM announcement claims are based on LSPR measurements calculated by averaging the ITR in several workloads.

LSPR measurements

There are three types of LSPR measurements:

- ▶ Actual processor measurements
- ▶ Projections

Primary models of each processor series are always measured for the LSPR. It is impractical, however, for IBM to allocate the necessary resource to measure every individual processor model announced. For those that are not measured, ITR numbers are projected. A projected ITR is based on the known processor internal design, and on the known characteristics of the subject workload on similarly designed processor models.

Projections have been shown to have accuracy comparable to that of measurements, where subsequent testing has occurred.

► Estimates

Many older processor models (both IBM and IBM-compatible) measured in the past are no longer accessible for testing. However, it is highly desirable to maintain these processors in LSPR tables for system control programs (SCPs) that are supported. Therefore, as the LSPR moves ahead to more current software levels, ITRs for these processors must be estimated. Estimates are based on known deltas between the older software and the current software on similarly designed processor models.

As capacities of today's processors grow ever larger, it becomes more difficult to commit the time and external resources necessary for full, unconstrained LSPR measurements. Experience has shown that there are estimation techniques, based on making reduced resource measurements and analysis, that can yield reliable results. Estimated ITRs are considered to have an accuracy very close to that of actual measurements. Estimates are not provided in the LSPR without a high degree of confidence in the process, and in the resulting ITR values.

For more information and LSPR tables you can access the following site:

<http://www-1.ibm.com/servers/eserver/zseries/lspr/>

1.13 LSPR workloads

- ❑ z/OS
 - ODE-B
 - CB-L commercial batch long job steps (CBW2)
 - WASDB WebSphere app+DB (Trade2-EJB)
 - OLTP-W Web-enabled OLTP (CICS/DB2 with web front-end)
 - OLTP-T Traditional OLTP (IMS)
 - LoIO-Mix, a mixing of CB-L, WASDB and OLTP-W
 - WEB Mixed
- ❑ z/VM
 - WASDB/LVm Linux guests with WebSphere app+DB (Trade2-EJB)
- ❑ Linux on System z
 - WASDB/L WebSphere app+DB (Trade2-EJB)
 - EAS-AS/L (est) EAS app servers (SAP SD benchmark)

Figure 1-13 LSPR workloads

Large Systems Performance Reference (LSPR)

IBM uses the LSPR benchmark for System z processors. When upgrading or buying a new machine, capacity planners should always consider using LSPR tables in order to figure out the possible model based on LSPR MIPS. The tools zPCR and zCP3000 are based on the same LSPR benchmark MIPS and the Internal Throughput Rate Ratios from the tables.

The IBM Large Systems Performance Reference (LSPR) ratios represent IBM's assessment of relative processor capacity in an unconstrained environment for the specific benchmark workloads and system control programs specified in the tables. Ratios are based on measurements and analysis. The amount of analysis as compared to measurement varies with each processor.

LSPR workloads

There are workloads for z/OS, z/VM, and Linux on System z. LSPR recently implemented new workloads to cope with the evolution of data processing types currently performed in mainframe clients. As a consequence, the same machine may have different numbers for MSU/hour, MIPS and RPPs.

Other workloads have been created on top of new basic workloads; there is a mixing of the basic ones. Examples include LoIO-Mix (CB-L, WASDB and OLTP-W) and WEB mixed. In the following section we describe the CB-L workload in greater detail.

CB-L (formerly CBW2) - Commercial Batch Long Job Steps

The CB-L workload is a commercial batch jobstream which is reflective of large batch jobs with fairly heavy CPU processing. Each copy consists of 22 jobs, with 157 job steps. These jobs are more resource-intensive than jobs in the CB-S workload, use more current software, and exploit ESA features. Sufficient copies of the workload are used to achieve high processor utilization. The work performed by these jobs includes various combinations of C, COBOL, FORTRAN, and PL/I compile, link-edit, and execute steps. Sorting, DFSMS utilities (for example, dump/restore and IEBCOPY), VSAM and DB2 utilities, SQL processing, SLR processing, GDDMgraphics, and FORTRAN engineering and scientific subroutine library processing are also included.

Compared to CB-S, there is much greater use of JES processing, with more JCL statements processed and more lines of output spooled to the SYSOUT and HOLD queues. This workload is heavily DB2-oriented, with about half of the processing time spent in performing DB2-related functions.

Measurements

Measurements are made with z/OS, OS/390, DFSMS, JES2, RMF, and RACF. C/370, COBOL II, DB2, DFSORT, FORTRAN II, GDDM, PL/I, and SLR software is also used by the job stream. Access methods include DB2, VSAM, and QSAM. SMS is used to manage all data. Performance data collected consists of the usual SMF data, including type 30 records (workload data), and RMF data.

The CB-L job stream is replicated to assure a reasonable measurement period, and the job queue is preloaded. A sufficient number of initiators are activated to ensure a high steady-state utilization level, approaching 100%. The number of initiators is generally scaled with processing power to achieve comparable tuning across different machines. The measurement is started when the job queue is released, and ended as the last job completes. Each copy of the workload uses its own data sets, but jobs within the workload share data.

1.14 Benchmark methods

- ❑ **LSPR concerns to create a valid benchmark:**
 - Assure adequate configuration (storage, channels, DASD)
 - Distribution of system data
 - Distribution of program libraries
 - Distribution of data files (datasets) and databases
 - Files and databases restored to pristine state
 - Logon users (terminals or clients), or load job queue
 - Determine measurement period to obtain a repeatable sample of work
 - Adjust activity to realize target processor utilization level
 - Assure steady state has been achieved
 - Capture appropriate performance monitor data
 - Capture operator console logs
 - Verify that no hardware errors occurred
 - Verify measurement data against acceptance criteria
 - Construct detailed measurement reports

Figure 1-14 Benchmark methods

IBM Large Systems Performance Reference (LSPR) benchmark

The IBM Large System Performance Reference (LSPR) ratios represent an IBM assessment of relative processor capacity in an unconstrained environment for the specific benchmark workloads and system control programs.

LSPR workload types

Each individual LSPR workload is designed to focus on a major type of activity, such as interactive, online database, or batch. The LSPR does not focus on individual pieces of work such as a specific job or application. Instead, each LSPR workload includes a broad mix of activity related to that workload type. Focusing on a broad mix can help assure that resulting capacity comparisons are not skewed. Figure 1-14 describes various concerns in performing benchmarks.

Using benchmarking for capacity planning

Benchmarking refers to running a real workload in a particular hardware configuration to measure CPU power.

Using benchmarking is often more effective than previous methods, but is also more expensive and time-consuming. Benchmarks are divided into two categories:

- ▶ A benchmark from others

With these benchmarks, the evaluations of standard workloads run in a controlled environment result in numbers that are used as a reference. Normally, consulting companies perform these studies.

- ▶ Your own benchmark

With these benchmarks, you run your own workload on a specific hardware configuration. Unlike the analytic and discrete methods, the benchmark requires executable code and installed hardware to predict how a certain combination of resources will perform in a specific situation.

Note: IBM Worldwide Client Benchmark Centers provide benchmark capability across the globe for IBM server and storage technology, including proof of concept, scaling, and performance services.

1.15 IBM capacity planning tools

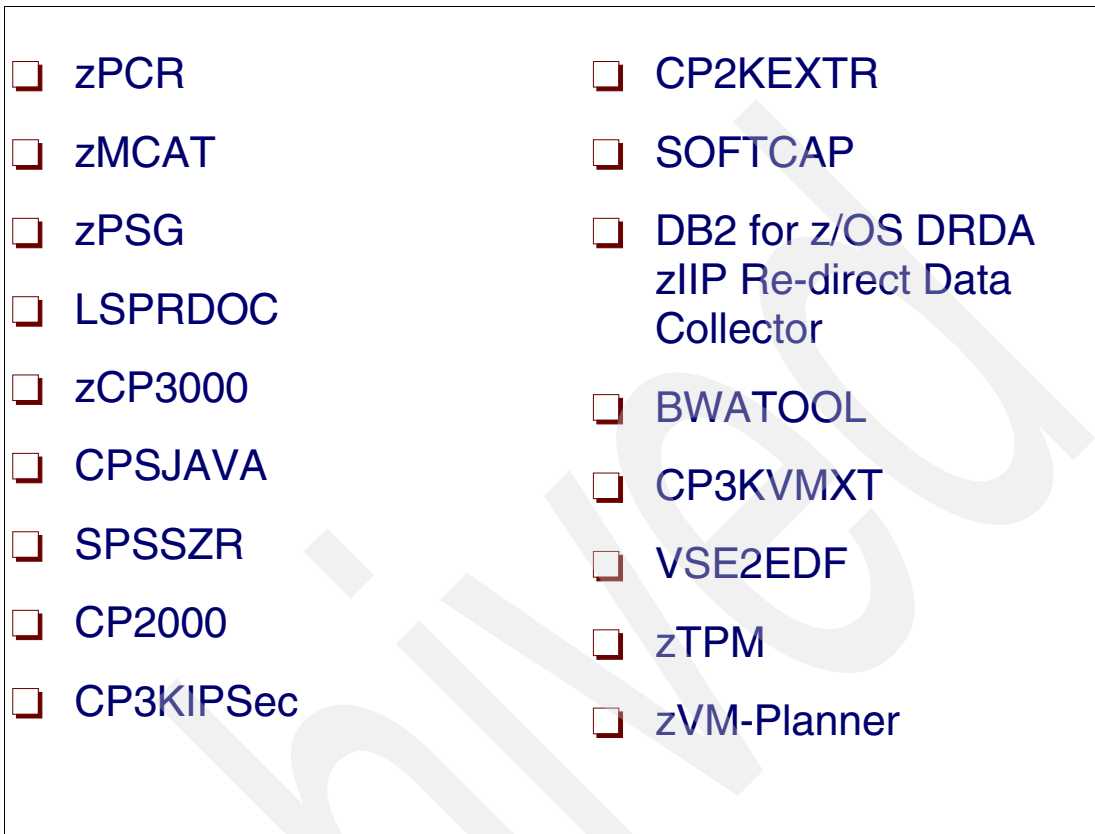


Figure 1-15 IBM capacity planning tools

IBM capacity planning tools

IBM capacity planning tools are intended to be used by the IBM technical support team when supporting a client's capacity planners to achieve the best results in terms of resource measurement.

zPCR

zPCR is a PC-based tool that provides capacity planning insight for zSeries processors running various workload environments under z/OS, z/VM, and Linux. Capacity results are based on IBM LSPR data. In addition to the actual LSPR workloads, four predefined z/OS and up to five user-defined workload mixes are available for capacity planning purposes. Any of these workloads can be displayed in the LSPR tables, or defined to specific partitions. Capacity numbers displayed are relative to a user-selected reference CPU, whose capacity may be scaled to any desired metric.

zMCAT

zMCAT, the zSeries Migration Capacity Analysis Tool, can be used to compare the performance of production batch workloads before and after migrating the system image to a new processor.

zPSG

zPSG is a PC-based tool that estimates the IBM System z or IBM zSeries capacity requirement for various workload types. Results are shown in terms of the number of users or

transaction rates supported, or the projected utilization to support a fixed amount of work or fixed number of users. z/PSG currently supports the environment WebSphere Application Server environment.

LSPRDOC

LSPRDOC, the Large Systems Performance Reference, provides reliable relative capacity data for System/370, System/390, and z/Architecture processors. The LSPR provides sets of workload-sensitive ITRR tables and a methodology for their proper use to reliably determine relative processor capacity for production workloads.

zCP3000

zCP3000 is a Java PC-based tool designed to do performance analysis and capacity planning for IBM zSeries and S/390 processors running various SCP/workload environments. zCP3000 can also be used to analyze logically partitioned processors and DASD configurations.

Attention: zCP300 is an IBM internal tool only. zCP3000 uses zPCR data directly. zCP3000 uses zTPM. Run zCP3000 and go right into zTPM. zCP3000 defaults to the 90th percentile of the prime shift not peak. Contact the Capacity Planning Support (CPS) team at the IBM Washington Systems Center, Gaithersburg, Maryland, for more information about this topic.

CPSJAVA

CPSJAVA provides the IBM Java environment for your PC as required by various CPSTools. It is updated on a periodic basis as needed to stay with a supported level of IBM Java.

SPSSZR

SPSSZR was a PC-based tool that provided capacity planning insight for processors in a Parallel Sysplex environment. However, the tool's functions have now been incorporated into zCP3000.

CP2000

CP2000 is a PC-based productivity tool for performance analysis and capacity planning for IBM System z, IBM zSeries, and IBM-compatible S/390 processors running various SCP/workload environments. CP2000 can also be used to analyze logically partitioned processors and DASD configurations.

CP3KIPSec

CP3KIPSec is a new host productivity tool that uses a console interface to query statistics about the TCP/IP address space at regular intervals. The file produced is used by zCP3000. zCP3000 uses the data in the file to estimate the cost of using IPsec in terms of zIIP engines.

CP2KEXTR

CP2KEXTR is a host-based utility that reads SMF data from a z/OS system and produces input files for CP2000 or zCP3000, zMCAT, and Disk Magic.

SOFTCAP

SOFTCAP is a PC-based tool that evaluates the effect on z/Architecture and S/390 processor capacity when migrating to newer levels of software, including z/OS or OS/390, CICS and IMS. In addition, SOFTCAP can assess the effect on capacity when converting from 31-bit to

64-bit addressing, which is supported on zSeries processors by z/OS and OS/390 V2R10. Results are presented in the form of tables.

DB2 for z/OS DRDA zIIP Re-direct Data Collector

DB2 for z/OS DRDA zIIP Re-direct Data Collector uses DB2 Accounting 101 records (1 and 2) to estimate potential zIIP offload percentages for DB2 V7 and V8.

BWATOOL

BWATOOL is a host-based tool that analyzes SMF data for batch jobs and estimates how long jobs would run on a target system. Analysis is performed at a job level or step level.

CP3KVMXT

CP3KVMXT is a host-based utility that reads saved or real-time CP Monitor data from a z/VM system and produces an input file for CP2000 or zCP3000.

VSE2EDF

VSE2EDF is a workstation utility that reads CPUMON files created on a z/VSE system and converts them to standard EDF files for input into zCP3000.

zTPM

zTPM, the Tivoli Performance Modeler, is a PC-based tool for building a model of a z/OS-based system, and then running various “what if” scenarios. zTPM uses simulation techniques to model the impact of changes on individual workload performance.

Note: This installation of zTPM requires an install password that is only disclosed during zTPM training sessions.

zVM-Planner

zVM-Planner is a PC-based productivity tool under Windows, designed to provide capacity planning insight for IBM System z processors running various Linux applications as guests under z/VM. Capacity results are based on analysis of a variety of benchmarks and analysis techniques. The tool is generic concerning software release levels, and generally applies to z/VM v5.1 and later.

1.16 Resource Management Facility overview

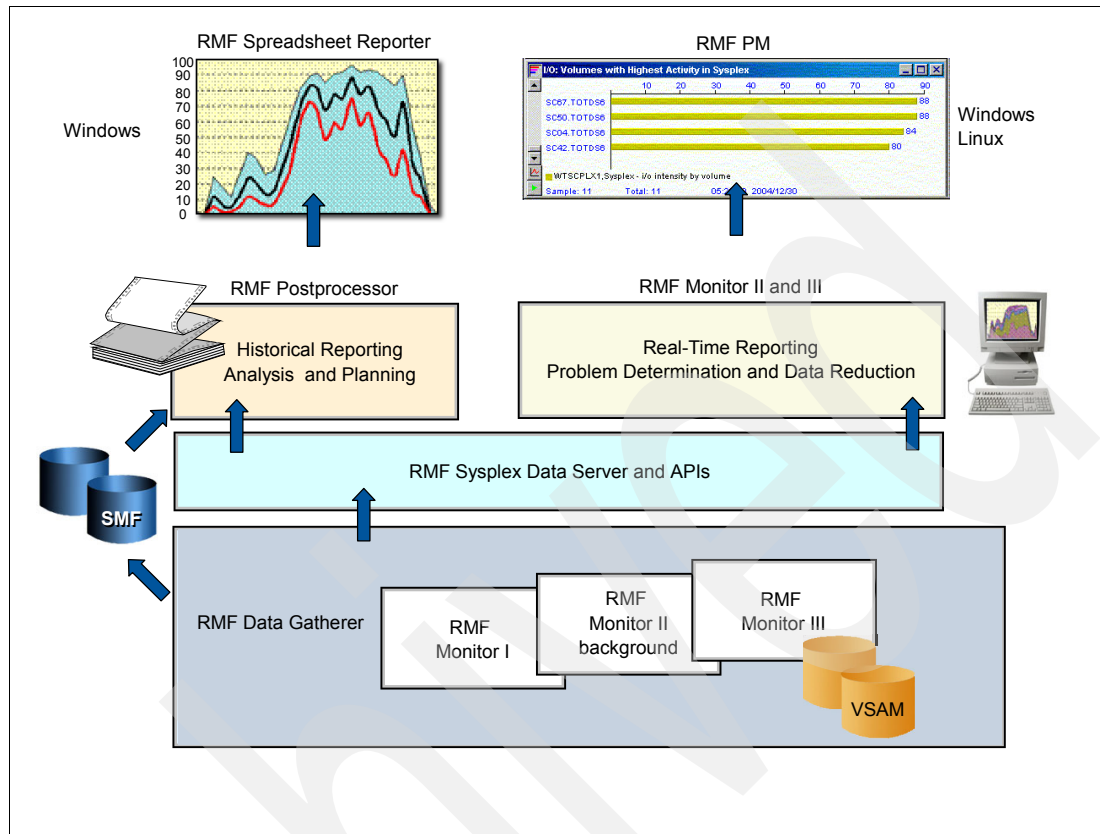


Figure 1-16 Resource Management Facility as a capacity planning information tool

Resource Management Facility overview

z/OS Resource Measurement Facility (RMF) is an optional element of z/OS. It is a product that supports installations in performance analysis, capacity planning, and problem determination. For these disciplines, different kinds of data collectors are needed:

- ▶ Monitor I long-term data collector for all types of resources and workloads. The SMF data collected by Monitor I is mostly used for capacity planning, but is also used performance analysis. Additionally, there are products like Tivoli Decision Support that use SMF records gathered by Monitor I.
- ▶ Monitor II snapshot data collector for address space states and resource usage.
- ▶ Monitor III short-term data collector for problem determination, workflow delay monitoring, and goal attainment supervision. This data is used by RMF PM Java Client, Tivoli DM and Tivoli TBSM.
- ▶ Data collected by all three gatherers can be saved persistently for later reporting.
- ▶ Monitor II and Monitor III are online reporters. Monitor I and Monitor III can store the collected data long term.

To learn more about RMF, visit the RMF home page:

<http://www.ibm.com/servers/eserver/zseries/zos/rmf>

Regardless of the method or tool used, RMF data is a complete source of information for resource capacity planning. See 3.1, "Resource Measurement Facility overview" on page 176 for more information about RMF.

1.17 zCP3000 introduction

- ❑ Automated input
- ❑ Performance analysis and capacity planning for:
 - Processor
 - Workload
 - Storage
 - DASD
- ❑ Parallel Sysplex evaluation
- ❑ Sampling what you can do...

Figure 1-17 zCP3000 introduction

zCP3000 introduction

zCP3000 is a sophisticated tool that IBM technical sales specialists execute on behalf of clients. It incorporates the IBM best understanding of the factors that will determine the capacity that a particular hardware configuration will deliver. The number of engines, the number of LPARs, the mix of special engine types, the impact of N-way effect, different workload characteristics and many other factors have been accounted for in the mathematical model contained within zCP3000.

zCP3000 is an internal IBM tool that analyzes data from the client and generates reports that show a kind of “snapshot” of main resource utilization. It provides performance analysis and capacity planning information from SMF registers. zCP3000 requires an input that is extracted from the SMF files by the CP2KEXTR program data extractor. It is one of a family of tools produced and maintained by the Capacity Planning Support (CPS) team at the IBM Washington Systems Center, Gaithersburg, Maryland.

zCP3000 data extractor

IBM Technical Support teams and Business Partners use the data extractor. A support manual for CP2KEXTR, available from IBM, explains how to customize the batch process of extraction. Generally, this task is executed at the client site; a Technical Support Analyst from IBM or a Business Partner demonstrates the process to the client Technical Support Analyst. The client analyst then selects the best period to analyze with the zCP3000. Normally, the periods chosen are peak periods. zCP3000 defaults to the 90th percentile of prime shift not peak.

zCP3000 documentation

The zCP3000 documentation describes the amount of extracted data necessary to achieve consistent analysis results. After extraction, the IBM Technical Support Analyst runs zCP3000 with the collected input. Prior to extraction, however, it is important that both IBM analysts and client analysts understand what is required in terms of analysis, projections, and simulations. zCP3000 will extract statistics regarding the data from the dates and times chosen, but the client decides which workload growth percentages and which periods to use in forecasting analysis (IBM Technical Support may make suggestions for a case study analysis).

Note: The zCP3000 documentation, *zCP3000 User's Guide*, Document ID PRS1772, is available at the following site:

<http://www.ibm.com/support/techdocs>

zCP3000 analysis

zCP3000 analysis, a task which involves complete teamwork, helps clients to understand and manage their resources (such as processors, storage, DASD I/O) and features (such as Parallel Sysplex).

Note: For capacity planning, zCP3000 has superseded the S/390 Parallel Sysplex Quick-Sizer (SPSSZR) tool produced by the Washington System Center (WSC). zCP3000 can project the overall hardware requirements, including the following items:

- ▶ Number of z/Architecture™ systems required for the workload
- ▶ Processor utilization
- ▶ Storage requirement
- ▶ Number of Coupling Facilities
- ▶ Coupling Facility utilization
- ▶ Number of Coupling Facility links
- ▶ Average Coupling Facility link utilization
- ▶ Processor link utilization
- ▶ Processor link mode
- ▶ CPU service/response time
- ▶ zCP3000 supports CF Duplexing and the new XCF algorithms

1.18 zCP3000 functions

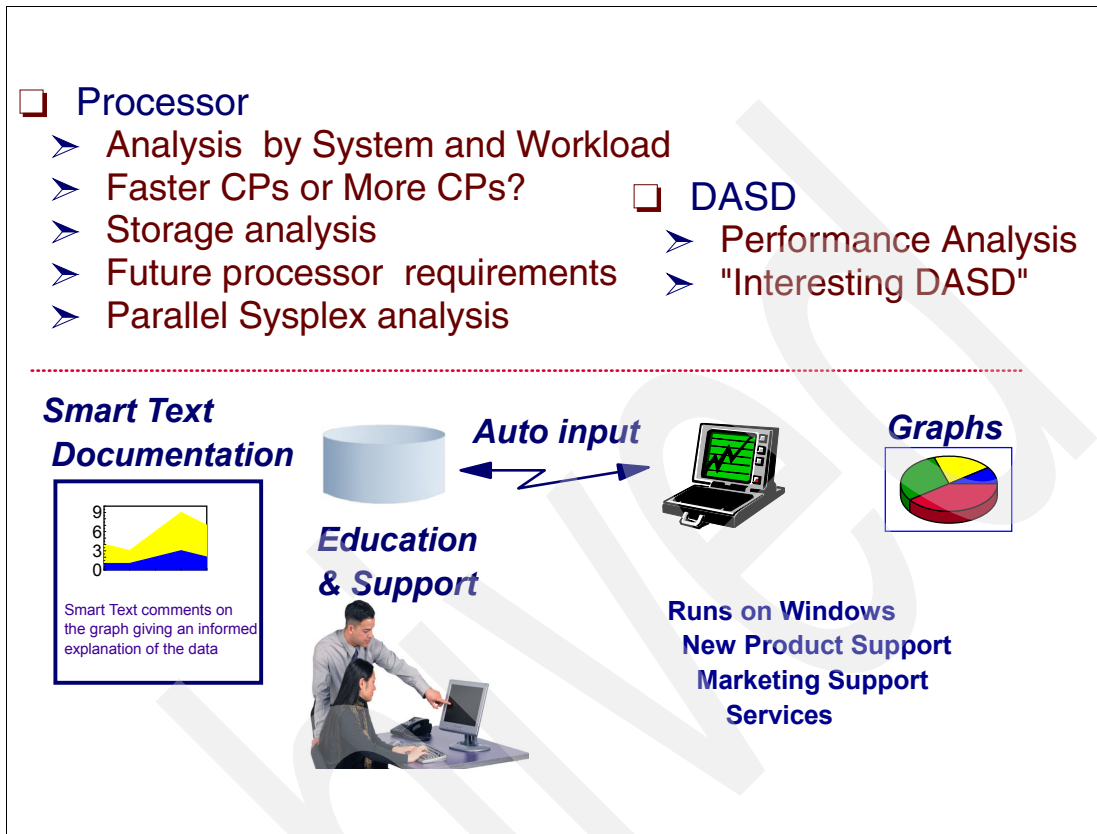


Figure 1-18 zCP3000 functions

zCP3000 functions

zCP3000 is a PC-based productivity tool that runs under Windows XP or Vista and is designed to provide performance analysis and capacity planning for IBM zSeries and S/390 processors running various SCP or workload environments. zCP3000 can also be used to analyze logically partitioned processors and DASD configurations. Input, which is usually RMF data from the processors considered for replacement, can be automated using the separate tool CP2KEXTR.

Future processor requirements

When considering future processor requirements, you must keep in mind IT budgets and software expenditures regarding acquiring additional processors. zCP3000 provides functions that will help you plan with these considerations in mind.

Analysis by system and workload

zCP3000 can help your capacity planning by showing processor statistics such as the average, 90tile (ninety percentile), and peak-to-average ratios of CPU consumption. zCP3000 defaults to the 90th percentile of prime shift not peak. The scope can be the entire machine, the LPARS, the workloads, and Coupling Facility utilization.

Faster CPs, or more CPs

zCP3000 can help answer one of the most important questions that capacity planners ask, which is: are *faster* CPs, or *more* CPs, needed? An understanding of the unpredictable peaks

of CPU consumption from e-business workloads, and the behavior of these workloads during the CPU life cycle, is another benefit of using zCP3000.

Storage analysis

Storage analysis is also extremely important because no guidelines or rules of thumb exist for all cases, and zCP3000 can help with this challenge as well. It is essential to obtain a correlation between your CPU consumption and real storage utilization. A basic question to be answered in capacity planning is, when your machine achieves the maximum acceptable CPU for your workloads, what is the amount of real storage necessary to satisfy this correlation?

Parallel Sysplex analysis

For more detailed information regarding capacity planning considerations for Parallel Sysplex, see *OS/390 MVS Parallel Sysplex Capacity Planning*, SG24-4680.

1.19 Balanced systems analysis

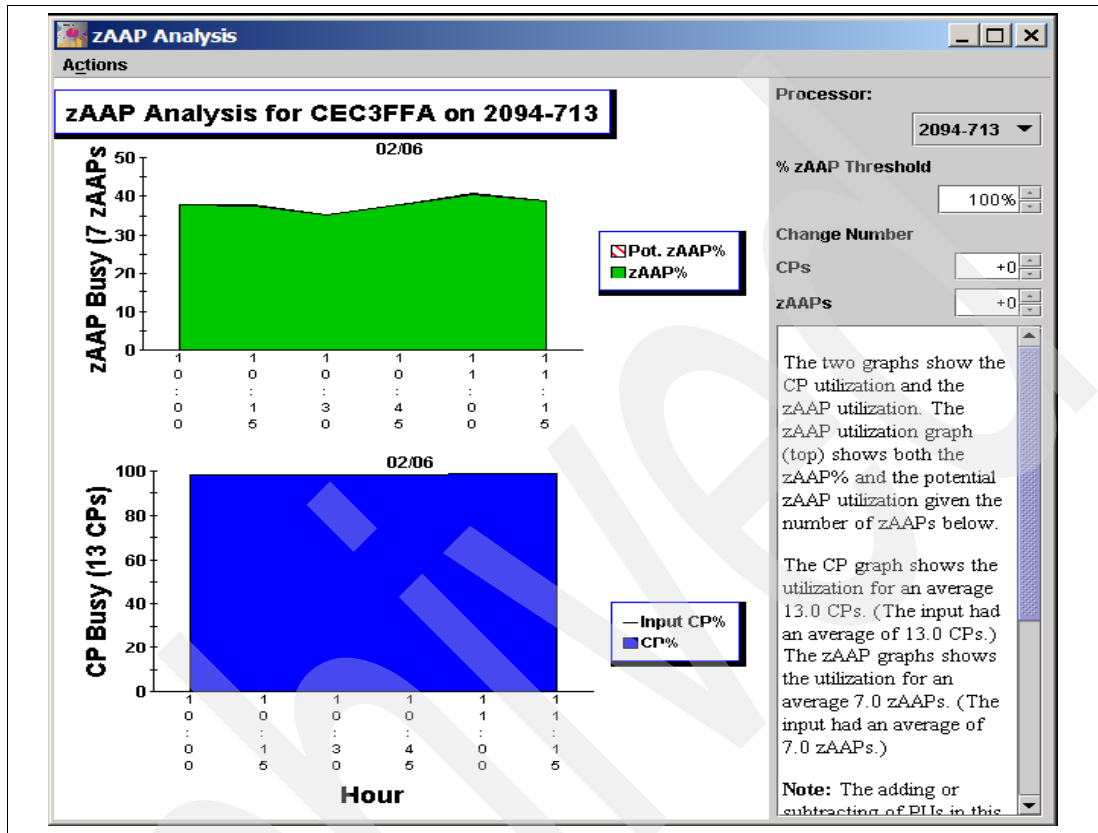


Figure 1-19 Balanced systems analysis

Balanced systems analysis

All zCP3000 analysis is comprised of both graphs and text. The text not only describes what the graph means, but also puts it in terms of the actual values in the graph. The text, called Smart Text, also makes intelligent suggestions. The graph in Figure 1-19, for example, shows the “zAAP Analysis” panel, where the top graph shows the number of physical zAAPs configured, as part of the y-axis title. The graph plots the actual amount of zAAP busy in this model along with the amount of potential zAAP busy (as a busy on the number of zAAPs). The analysis assumes that zAAP potential will run on a zAAP, if present.

zCP3000 graphs

zCP3000 provides an in-depth understanding of the graphic and Smart Text analysis. Using zCP3000 can help avoid unnecessary I/T costs in terms of resource acquisition, and help capacity planners to highlight the real point of the resource acquisition. IBM offers training support regarding the best utilization and interpretation of zCP3000 output reports.

1.20 Capacity planning tools

❑ zCP3000 is included in a suite of tools (listed here) that are supported by the CPS team:

❑ zCP3000

❑ SMF Type 30 Analysis (BWATOOL)

❑ zMCAT

❑ zPCR

❑ zPSG

❑ SoftCap

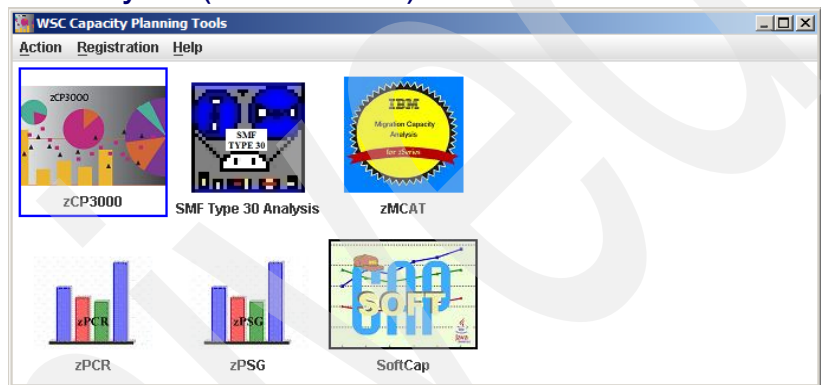


Figure 1-20 Capacity planning tools

Capacity planning tools

zCP3000 is included in a suite of tools that are supported by the capacity planning support (CPS) team. The WSC Capacity Planning Tools facility has been designed to house each of these tools so they are accessible from one location. To learn about any of these tools, right-click an icon (see Figure 1-20) and click Detail. Or, highlight an icon and click Action and Detail.

SMF Type 30 analysis (BWATOOL)

The Batch Workload Analysis Tool (BWATOOL) runs as an MVS job against a client's SMF data. It analyzes SMF data to determine how long jobs might run on a proposed System z processor. It reports CPU time for individual jobs on the current processor and assesses processing time on planned new processors for individual jobs and job steps.

IBM System z Migration Capacity Analysis Tool

The IBM System z Migration Capacity Analysis Tool (zMCAT) compares the performance of production workloads and provides the following functions and capabilities:

- ▶ Verify sizing
- ▶ Based on SMF data
- ▶ Collected before and after migration
- ▶ Uses CP2KEXTR (Type 30 or 110 records)
- ▶ Uses CP2KCOVG to group and filter by j/s/n or trans/appl
- ▶ Removes work with samples that are too small

- ▶ Removes work with too much variance
- ▶ Automated analysis of individual stable jobs or transactions
- ▶ Sort and produce results on your PC

Processor Capacity Reference for IBM System z

Processor Capacity Reference for IBM System z (zPCR) is a LSPR function that provides processor capacity comparisons and capacity ratios for up to 10 different SCP/workload environments. Specific workloads and the order presented is user controlled.

Processor Selection Guide for IBM System z

Processor Selection Guide for IBM System z (zPSG) performs processor sizing for new applications. It is based on high-level characterizations of transactions or users and is intended to provide sizing estimates for the following environments:

- ▶ Application environments currently supported
- ▶ WebSphere Application Server for z/OS and Linux
- ▶ Apache Webserver on Linux
- ▶ WebSphere Portal for z/OS and zLinux

Software Migration Capacity Planning Aid

Software Migration Capacity Planning Aid (SoftCap) evaluates capacity effects expected due to software upgrades for z/OS for CICS, IMS, batch, DB2, TSO, and system work.

SoftCap is downloadable from the following Web sites:

- ▶ Clients
<http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS268>
- ▶ Business Partners
<http://www.ibm.com/partnerworld/sales/systems>

Run SoftCap to determine your expected increase in CPU utilization (if any) and to identify your storage requirements, such as how much storage is needed to IPL. For reference information see the SoftCap User's Guide that is provided with the tool.

1.21 zCP3000 summary

- ❑ Delivers performance analysis and capacity planning at the level you need
- ❑ Graphs and smart text, which guide you in the analysis
- ❑ Deliverables include:
 - Foils
 - Automated documentation that can be customized for a focused deliverable
 - Flexibility to import the document into Word Pro, Word, or an Internet browser
- ❑ And there's much more available:
 - Analytic processor response time analysis
 - A first step in transaction simulation
 - Parallel Sysplex migration and analysis
 - New workload capacity requirements

Figure 1-21 zCP3000 summary

zCP3000 summary

All zCP3000 functions are designed to support the capacity planning and performance demands of IBM clients. The cornerstone of zCP3000 is the experience of the IBM team, helping clients to create efficient ways to measure and predict resource consumption.

With zCP3000 analysis, clients can minimize the discrepancies between real-life performance and predictions. Some clients use zCP3000 analysis reports to compare with their own capacity planning reports.

zCP3000 functions include analytic processor response time analysis, basic transaction simulation, Parallel Sysplex migration and analysis, and new workload capacity requirements. And not only is zCP3000 a powerful capacity planning and performance tool, but you can also use its outputs in presentations and support documents. zCP3000 will continue to be improved to support the efforts of capacity planners.

1.22 IBM Processor Capacity Reference for zSeries (zPCR)

- ❑ zPCR is a PC-based productivity tool under Windows
 - A Java-based PC tool
- ❑ Clients can plan their entire zSeries environment:
 - Including adding specialty processors such as zAAPs, ICFs, and IFLs
 - Designed to provide capacity planning insight for IBM System z processors running various workloads
 - Environments include z/OS, z/VM, and Linux
 - Capacity results are based on the IBM LSPR data supporting all IBM System z processors

Figure 1-22 zPCR tool

zPCR tool

zPCR is a PC-based productivity tool that runs under Windows XP or Windows 7. It is designed to provide capacity planning insight for IBM mainframe processors running various SCP/workload environments and using various LPAR configurations. Capacity results are based primarily on IBM published LSPR data.

Plan a zSeries environment

zPCR is available as a no-cost download available from IBM. The tool will also help IBM System z clients plan their entire IBM System z environment, including the addition of specialty processors such as zAAPs, ICFs, and IFLs. zPCR can do the complete job of estimating the impacts of adding a zAAP processor to the configuration, the impacts of adding CF engines, and the impacts of adding Linux or VM partitions using IFLs.

This tool gives capacity planners new capabilities to evaluate and size proposed IBM System z configurations, and allows capacity planners to perform the following actions:

- ▶ Identify the relative capacity relationships among different IBM System z hardware configurations
- ▶ Allow the user to enter an LPAR environment (either proposed or current) and determine the impacts on capacity of the specific LPAR environment

Supported hardware

The LPAR host processor is selected and configured with CPs, zAAPs, zIIPs, IFLs, and ICFs, where appropriate. Then each logical partition is defined, specifying type (CPU, IFL, or ICF), SCP/workload, LP configuration (dedicated or shared with the number of PUs), and weight and capping assignments. zAAP and zIIP logical PUs are always associated with a z/OS V1R6 (or later) logical partition.

General purpose CPs, zAAPs, zIIPs, IFLs, and ICFs, where they can be legitimately configured to the hardware, are fully supported.

Note: Due to the absence of supporting measurement data, be aware of the following points:

- ▶ A limit of 8 zAAP and 8 zIIP logical CPs is currently imposed for each z/OS logical partition.
- ▶ Capacity results cannot be derived when both zAAPs and zIIPs are included with a single z/OS logical partition. Two levels of LPAR Configuration Capacity Planning function are available.

Capacity planning for System z processors

Capacity ratios are provided on General Purpose CPs for any workload or workload mix associated with the currently selected LSPR table. Capacity for IFLs is available only on IBM System z processors and only for native Linux or z/VM with Linux guests, because z/OS cannot be run on an IFL.

LSPR data support

z/OS measurements are version V1R9, supporting up to 64 CPs, and version V1R8, supporting up to 32 CPs. These tables include LSPR workloads run under z/VM, and Linux as well. Each version of LSPR data carries five unique MVS workload primitives, as well as workloads for z/VM and Linux.

zPCR functionality

Two main functions are provided by zPCR:

- ▶ LSPR capacity ratio tables

This function provides capacity ratios for up to 10 SCP/workload environments displayed side by side. Specific workloads and the order presented is user controlled. Capacity ratios are provided on General Purpose CPs for any workload or workload mix associated with the currently selected LSPR table. Capacity for IFLs is available only on IBM System z processors and only for native Linux or z/VM with Linux guests.

- ▶ LPAR Configuration Capacity Planning

This function is designed to project the capacity expectation for any specific LPAR configuration on any specific LPAR host configuration. Capacity results are provided for each individual logical partition and for the LPAR host as a whole.

1.23 Getting started with zPCR

- ❑ Download the zPCR tool
- ❑ Get training materials (prerecorded lectures and handouts)
 - Print a copy of the User's Guide
- ❑ Obtain information about how to request defect support for zPCR
- ❑ Information about zPCR registration
- ❑ Prepare inputs to zPCR

Figure 1-23 Getting started with zPCR

Getting started with zPCR

Processor Capacity Reference for IBM System z (zPCR) is one of a family of tools produced and maintained by your Capacity Planning Support (CPS) team, part of IBM Advanced Technical Skills (ATS) in Gaithersburg, Maryland.

Note: The currently supported environment for running zPCR is Windows XP SP3. Windows 7 is also fully supported. zPCR can be expected to function under Windows 2000 and under Vista SP2; however, reported problems will be addressed only if they can be recreated under Windows XP.

Downloading zPCR and materials

IBM clients can obtain zPCR through the Internet at:

<http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS1381>

A zPCR User's Guide is distributed in Adobe Acrobat PDF format. It has been created to be compatible with Acrobat Reader 5.0 and later. The current release of Acrobat Reader for Windows is available at no charge from:

<http://www.adobe.com>

The related online help consists of the User's Guide text, implemented as Java Help. This help is context-sensitive to the window currently being viewed.

zPCR defect support

The zPCR tool, which is downloadable and usable by customers, is provided on an “as is” basis. Support is provided by the IBM Washington Systems Center (WC), including training materials and both “how to” support and FAQ support for the tool. The WSC provides defect support for the tool.

zPCR registration

A user registration process has been implemented to assist in monitoring the distribution and use of zPCR. Registration is required for continued usage. You must be connected to the Internet to register.

Until your registration process is completed, a registration form will appear each time zPCR is started. zPCR may be used up to three times without completing and successfully submitting the registration information. After that, the registration process must be completed before the function of the tool can be accessed.

Fill in the requested fields and click the Register (Internet) button. The primary value of providing a valid e-mail address lies in our (CPS Tools) ability to notify you of any critical news relating to zPCR usage and updates. Use of the e-mail address will be limited to this purpose only.

zPCR inputs

zPCR inputs include the Reference-CPU settings, including the scaling-factor and metric, and all LPAR Configuration Capacity Planning function definitions. Study files can be reloaded at a later time for review or further analysis. Using the menu-bar, you can save a study by clicking File, and then Save as (or Save, if this is already a named study). Save as will prompt for a file name. The default file extension assigned is zPCR.

Note: The following details related to zPCR briefly describe the tool’s capabilities and show various capabilities.

1.24 zPCR function selection

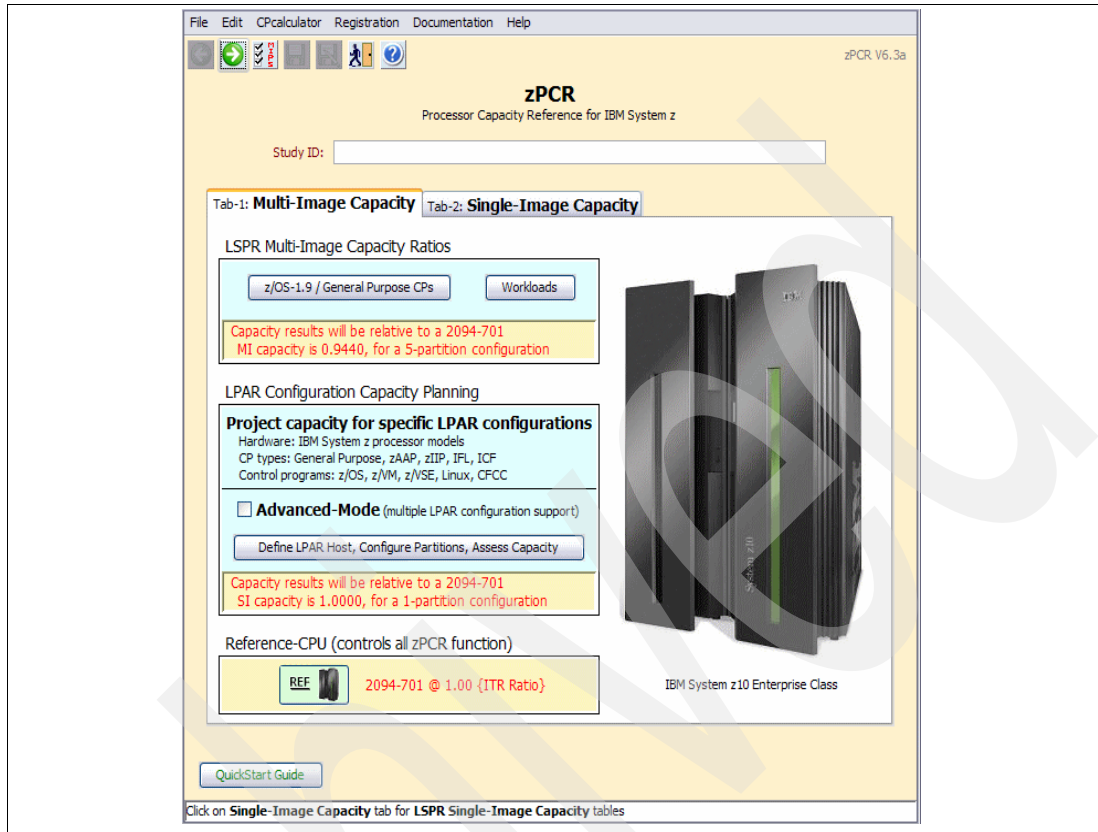


Figure 1-24 zPCR function selection

zPCR initiation

When zPCR execution is initiated, the Logo window is displayed. This window identifies the application and provides version and release information (this information is also available under **Help .. About zPCR** on each window's menu bar).

You must register to use zPCR. Upon initial use after install, you will be presented with a registration window requesting a few simple entries. The primary purpose of registration is to assure that you can be notified if a problem of serious concern occurs. Complete the form and click **Register**. After registration has successfully been completed, the registered user's name appears at the bottom of the Logo window.

zPCR function selection window

The Function Selection window is displayed immediately following the Logo window (after the Java code is compiled). This is the primary window used to access all the function of the tool. When you finish a zPCR function, control is returned to this window. From the Function Selection window a study can be saved, retaining most of the information that you have entered. You also terminate zPCR from this window.

Reference-CPU button

The Reference-CPU window is accessed primarily from the Function Selection window by clicking the Reference-CPU button, shown in Figure 1-24. There are two of these buttons, one on each of the Function Selection window tabs. Both buttons control the same object.

1.25 zPCR Reference-CPU

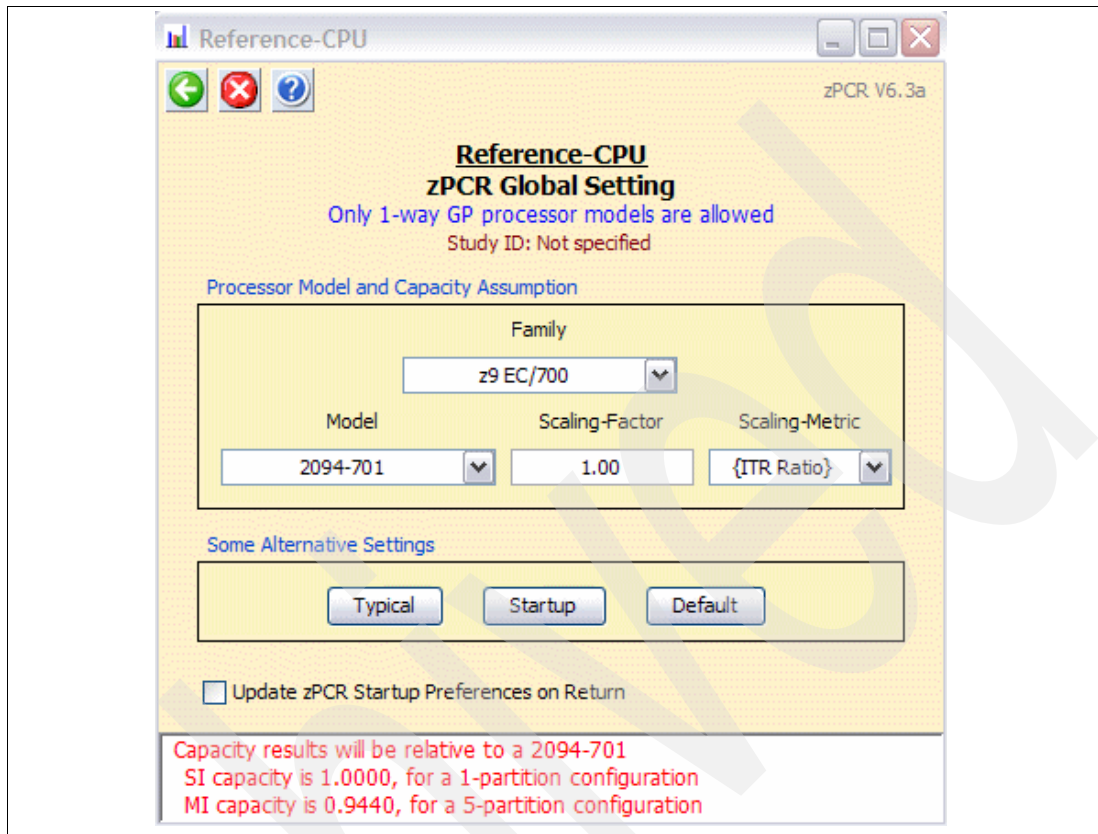


Figure 1-25 Single image reference-CPU screen

zPCR Reference-CPU

The Reference-CPU window is accessed primarily from the Function Selection window by clicking the Reference-CPU button. There are two of these buttons, one on each of the Function Selection window tabs. Both buttons control the same object.

LPAR Configuration Capacity Planning

The Reference-CPU can be set to any IBM System z processor 1-way model. Only models representing General Purpose CPs (as known to z/OS) can be selected.

The Reference-CPU model and its scaling-factor/metric can be changed at any time. When changed, capacity values in *all* of the LSPR Capacity Ratios tables and the LPAR Configuration Capacity Planning will be updated to reflect the change.

Attention: It is critical that all capacity results to be compared are obtained using a consistent Reference-CPU setting.

The LPAR host processor is selected and configured with General Purpose CPs, zAAPs, zIIPs, IFLs, or ICFs, where appropriate. Then each partition is defined, specifying type (GP, IFL, or ICF), SCP/workload, LP configuration (dedicated/shared with number of CPs), and weight/capping assignments. zAAP and zIIP logical CPs are always associated with a General Purpose z/OS V1R6 (or later) partition. General Purpose CPs, zAAPs, zIIPs, IFLs, or ICFs, where they can be legitimately configured to the hardware, are fully supported.

1.26 Choosing a workload mix

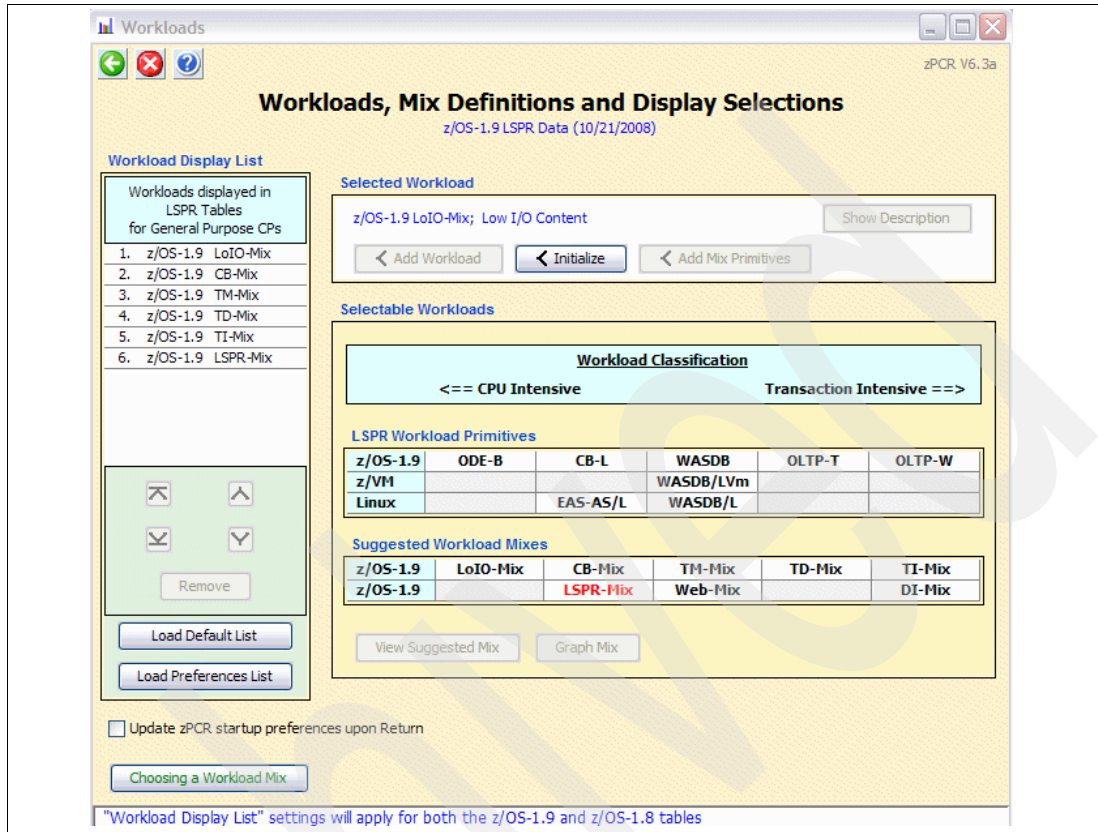


Figure 1-26 Choosing a workload mix

Choosing a workload mix

Figure 1-26, the Workloads window, is accessed from the Function Selection window by clicking the Workloads button (shown in Figure 1-24 on page 42) from either the Multi-Image Capacity tab or the Single-Image Capacity tab. After you arrive at this panel, select the Help icon (?) and Figure 1-27 on page 45 is displayed.

This panel can also be accessed through the Workloads button from any LSPR Processor Capacity Ratios table. Its purpose is to specify which SCP/workloads are to be displayed in the various LSPR Processor Capacity Ratios tables. Each LSPR table has its own associated workloads list.

Each workload is assigned a short name as well as a more descriptive long name. For the LSPR workload primitives, detailed descriptions are also available.

LSPR workloads

The Selectable Workloads group box, Figure 1-26, provides access to all the LSPR workload primitives and workload mixes associated with the currently active LSPR table. Clicking a workload will cause it to appear in the Selected Workload group box, where it can be directed to the Workload Display List group box. For workload primitives, the detailed description can be viewed by clicking the Show Description button.

1.27 zPCR choosing a workload mix Help panel

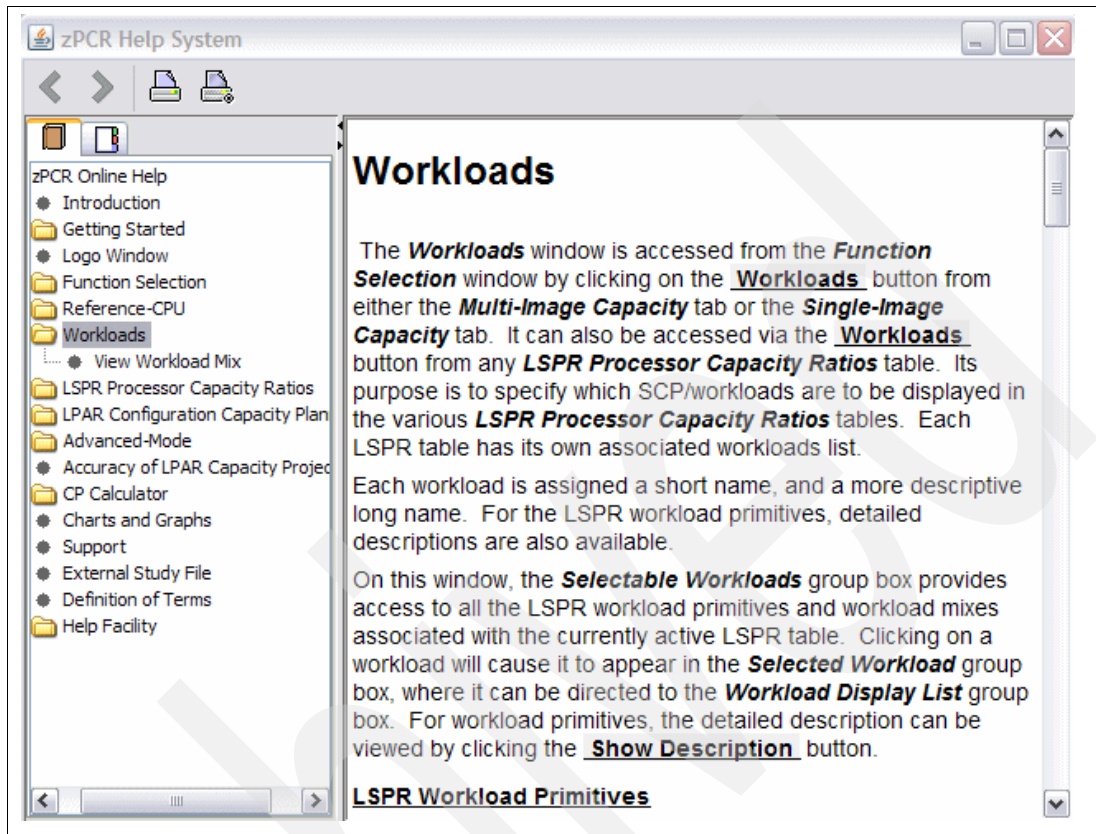


Figure 1-27 Choosing a workload mix

Choosing a workload mix

Next, you choose a mix of workloads that represent the most current of your workloads. zPCR offers extensive explanations of all the ZPCR used and available LSPR workloads.

Figure 1-27 shows a screen where the mix of workloads is described. The mix workloads are a mixture of the basic workloads such as WASDB, CB-L, CB-S, and OLTP-W.

Workload mix

The mix includes LoIO-Mix, CB-Mix, TM-Mix, TD-Mix, and TI-Mix. When bridging between LSPR versions, the use of any of these suggested mixes offers reasonable assurance that capacity results will be presented in a consistent manner.

Important: The workload named LSPR-Mix should not be used when bridging between LSPR tables, because its content is not consistent between the various tables.

LSPR workload mixes

In general, the MVS workload primitives should not be used directly to represent capacity for a production workload, because some of the z/OS workload primitives tend to stress hardware more than might be typical of production work.

MVS workload primitives are useful for representing the expected envelope (or bounds) of capacity expectation for each processor model.

Workload mixes based on the MVS LSPR workload primitives provide a better means for characterizing production workloads. Workload mixes allow more precise positioning in the potential capacity spectrum, and tend to smooth out any workload primitive anomalies that may exist.

Archived

1.28 LPAR Host and Partition Configuration panel

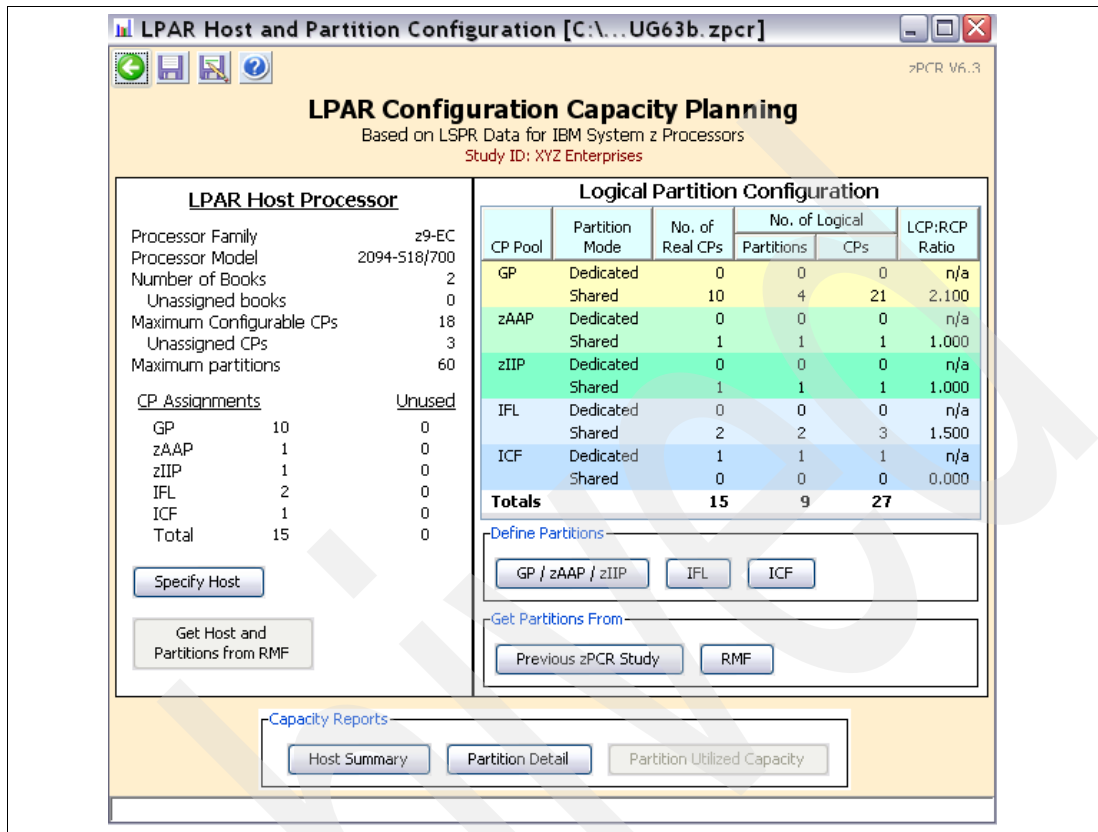


Figure 1-28 LPAR configuration capacity planning

LPAR Configuration Planning function

The LPAR Configuration Capacity Planning function provides capacity projections for a host processor with a specific LPAR configuration and with specific SCP/workload environments running in each partition. Capacity projections are generated for each partition as well as for the LPAR host as a whole.

This capability is accessed from the Function Selection window, shown in Figure 1-24 on page 42, by clicking the Define LPAR Host, Configure Partitions, Assess Capacity button. The LPAR Host and Partition Configuration window is displayed, from which the primary input and report windows are accessed.

Note: All capacity results for this function are based on the currently defined Reference-CPU. A 1-way processor model must be assigned.

Creating the LPAR configuration

The LPAR configuration is created using the following steps:

1. Define the LPAR host.

You must specify the LPAR host processor before you can define the partitions. Click the **Specify Host** button to display the LPAR Host Specification window (see LPAR Host Processor). Upon return, the LPAR host configuration is summarized on the left side of the LPAR Host and Partition Configuration window.

2. Define the partitions.

After you specify an LPAR host, you can define individual partitions by using buttons in the Define Partitions group box. Separate buttons are provided for General Purpose, IFL, and ICF partitions. zAAP and zIIP LCPs are always defined in conjunction with a General Purpose partition running z/OS V1R6 or later.

Note: Due to the absence of supporting measurement data, some restrictions are currently imposed on the number of zAAP and zIIP logical CPs that may be assigned to any individual GP partition.

The LSPR Single-Image Capacity Ratios tables are used exclusively to support this function (the multi-image tables play no role). The same Reference-CPU setting is used for both the LPAR Configuration Capacity Planning function, and for the LSPR Single-Image Capacity Ratios tables. Therefore, you can jump between these while continuing to observe capacity data on the same scale.

1.29 LPAR configuration capacity planning

Define General Purpose Partitions
Based on LSPR Data for IBM System z Processors
Study ID: XYZ Enterprises
z9-CC I lost = 2094-S10/700 with 15 CPs; GP=10 zAAP=1 zIIP=1 IFL=2 ICF=1
9 Active Partitions: GP=4 zAAP=1 zIIP=1 IFL=2 ICF=1

Include	Partition Identification				Partition Configuration						zAAP LCPs	zIIP LCPs
	No.	Type	Name	SCP	Workload	Mode	LCPs	Weight	Weight %	Capping		
<input checked="" type="checkbox"/>	1	GP	LP-01	z/OS-1.9*	LoIO-Mix	SHR	10	700	53.23%	<input type="checkbox"/>	0	0
<input checked="" type="checkbox"/>	2	GP	LP-02	z/OS-1.9*	CB-Mix	SHR	6	400	30.42%	<input type="checkbox"/>	1	0
<input checked="" type="checkbox"/>	3	GP	LP-03	z/OS-1.9*	TI-Mix	SHR	4	200	15.21%	<input type="checkbox"/>	0	1
<input checked="" type="checkbox"/>	4	GP	LP-04	z/VM	WASDB/Lvm	SHR	1	15	1.14%	<input checked="" type="checkbox"/>	n/a	n/a

Partition Summary by Pool

CP Pool	LPs	RCPs	DED LCPs	SHR		Sum of Weights
				LCPs	LCP:RCP	
GP	4	10	0	21	2.100	1,315
zAAP	1	1	0	1	1.000	400
zIIP	1	1	0	1	1.000	200
IFL	2	2	0	3	1.500	225
ICF	1	1	1	0	0.000	0

Input fields are white background; Single click selection field for drop-down list; Double click entry fields to open.

Figure 1-29 Defining general purpose partitions

LPAR configuration capacity planning

The Partition Definition window is accessed from the LPAR Host and Partition Configuration window, shown in Figure 1-28 on page 47, by clicking the GP button to define General Purpose partitions (and any associated zAAP and zIIP LCPs), the IFL button to define IFL partitions, or the ICF button to define ICF partitions.

Define General Purpose Partitions panel

Figure 1-29 shows the result produced by zPCR after you input the details of the current machine. The characteristics of the machine are listed here:

- ▶ It is a z9 EC with 15 CPs.
- ▶ Nine logical partitions are described at the top of the screen; the details shown are LP-01, LP-02, LP-03, and LP-04.

The primary table, shown at the top of Figure 1-29, provides an area where individual partitions are defined to the host. The host's configuration is shown on the top line above the primary table. The overall partition configuration, shown on the next line, is dynamically updated as partitions are added or modified.

The Partition Summary by Pool table, at the bottom of Figure 1-29, summarizes the current partition definitions for each CP pool. This table is updated dynamically as partitions are added or modified.

Note: From this panel you define the LPAR configuration and enter data representing each logical partition being considered in the study being done for your configuration. For more information about this topic, see the current version of *zPCR User's Guide* (version 6.3, February 9, 2010 at the time of writing).

Archived

1.30 zPCR graphics

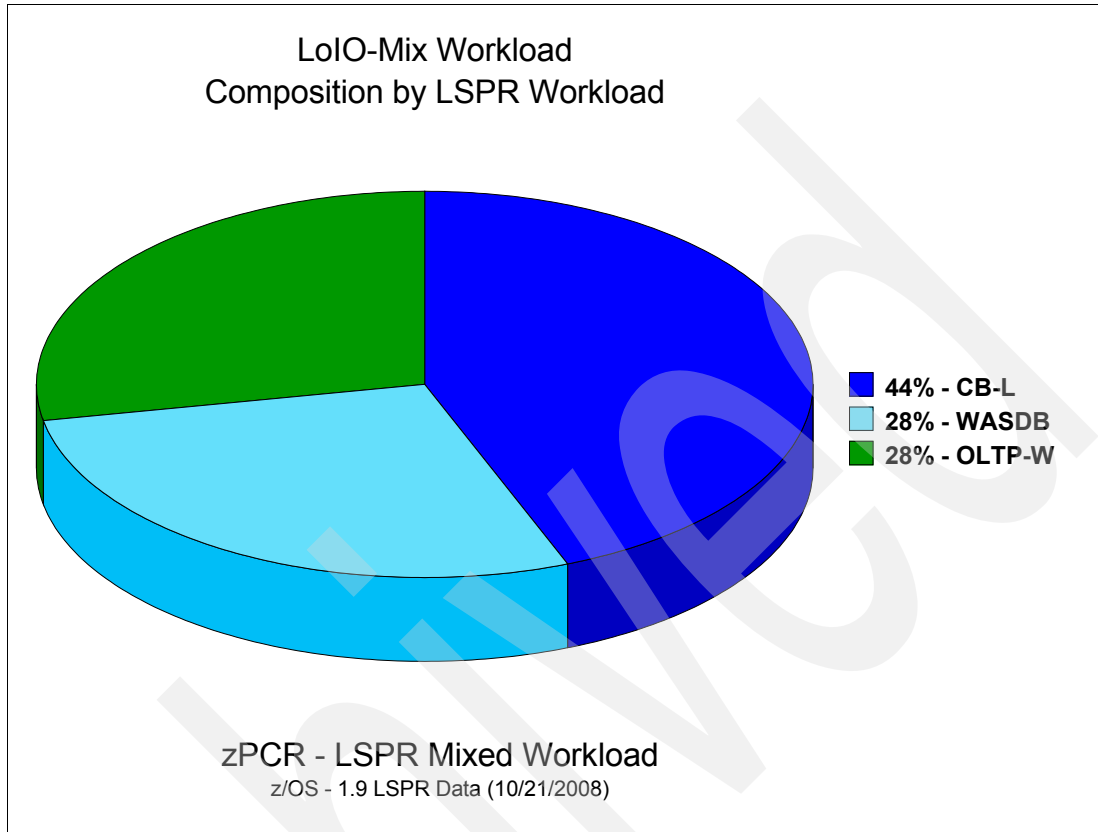


Figure 1-30 zPCR graphics

zPCR graphics

zPCR has also graphic capabilities. The pie chart in Figure 1-30 shows the calculated processor consumption per basic workload that forms the LoIO Mixed workload:

- ▶ CB-L is consuming 44%.
- ▶ WASDB is consuming 26%.
- ▶ OLTPW is consuming 29%.

Two graphs are available with this function, as explained here:

- ▶ Click the Capacity button for a bar graph showing the previous General Purpose capacity and the new zAAP environment capacity. The zAAP environment capacity is shown as a stacked bar, with separate values for the General Purpose CPs and for the zAAPs.
- ▶ Click Utilization for a line graph depicting utilization of the General Purpose CPs as utilization of the zAAPs declines. Dotted lines showing the currently set SDP values will appear if they fall within the utilization range plotted.

Additional graphs

Titles on each graph describe the intended purpose, and the bottom of the graph lists either the version of LSPR data or the zPCR version used to generate the graph, as well as the date when it was generated. All graphs relating capacity results will also include the Reference-CPU assumption that was used.

Graphs are available in the following functions:

- ▶ LSPR Multi-Image Capacity Ratios table
 - Bar graphs showing processor capacity for z/OS workloads
 - Bar graph showing processor capacity for all z/OS workloads
 - Line graphs showing CPU response time for z/OS workloads
- ▶ LSPR Single-Image Capacity Ratios tables
 - Bar graphs showing processor capacity by SCP/workload
 - Bar graph showing processor capacity for all workloads of an SCP
 - Line graphs showing CPU response time by SCP/workload
- ▶ Workloads window
 - Pie charts showing distribution of LSPR workload primitives assigned for mixes
- ▶ LPAR Configuration Capacity Planning function
 - Pie charts showing distribution of capacity by CP pool
 - Bar graphs showing processor capacity by CP pool
- ▶ zAAP Capacity estimator
 - Bar graph showing GP and zAAP capacity
 - Line graph showing GP utilization as zAAP utilization diminishes



Performance management

Performance management is a topic of great importance on all platforms. Measuring, comparing with standards, and tuning can reduce transaction response time and increase throughput, thereby reducing data processing costs.

This chapter provides an overview of the mechanisms used to measure and manage system performance. It covers the following subjects:

- ▶ Performance analysis introduction
- ▶ CPU and I/O performance metrics
- ▶ CPU-measurement Facility (CPMF)
- ▶ Formulas and laws that are applied to performance management

2.1 Service level agreement (SLA)

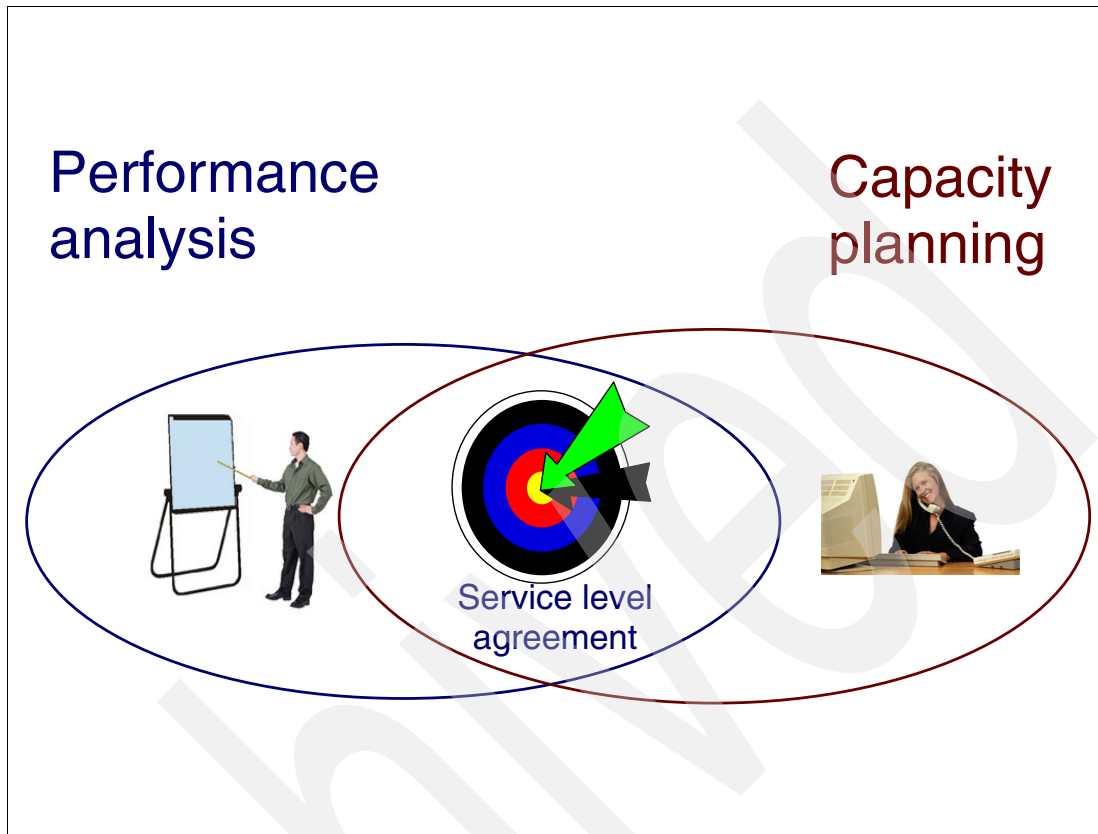


Figure 2-1 Service level agreement (SLA)

Service level agreement

A service level agreement (SLA) is a popular IT industry term for a documented agreement (sometimes literally a legal contract) between an IT organization (service provider) and a business organization (line of business, client). The term SLA emphasizes the formal nature of the service objective.

Performance analysis and capacity planning

The human view of a system's performance is often subjective, emotional, and difficult to measure. Because systems were created to meet the business needs of users, however, the concept of the service level agreement was introduced to match specified business requirements with subjective perceptions. The SLA is a contract that objectively describes and enforces such measurements as:

- ▶ Average transaction response time for network, I/O, CPU, or total. It is also best practice to correlate a response time SLA figure with a maximum transaction rate. A current trend is to have SLAs for DASD I/Os.
- ▶ The distribution of these response times (for example, 90% TSO trivial transactions at less than 200 milliseconds).
- ▶ System continuous availability metrics (24 by 7).

SLA is a fundamental step for performance analysis and capacity planning disciplines.

2.2 Performance analysis overview

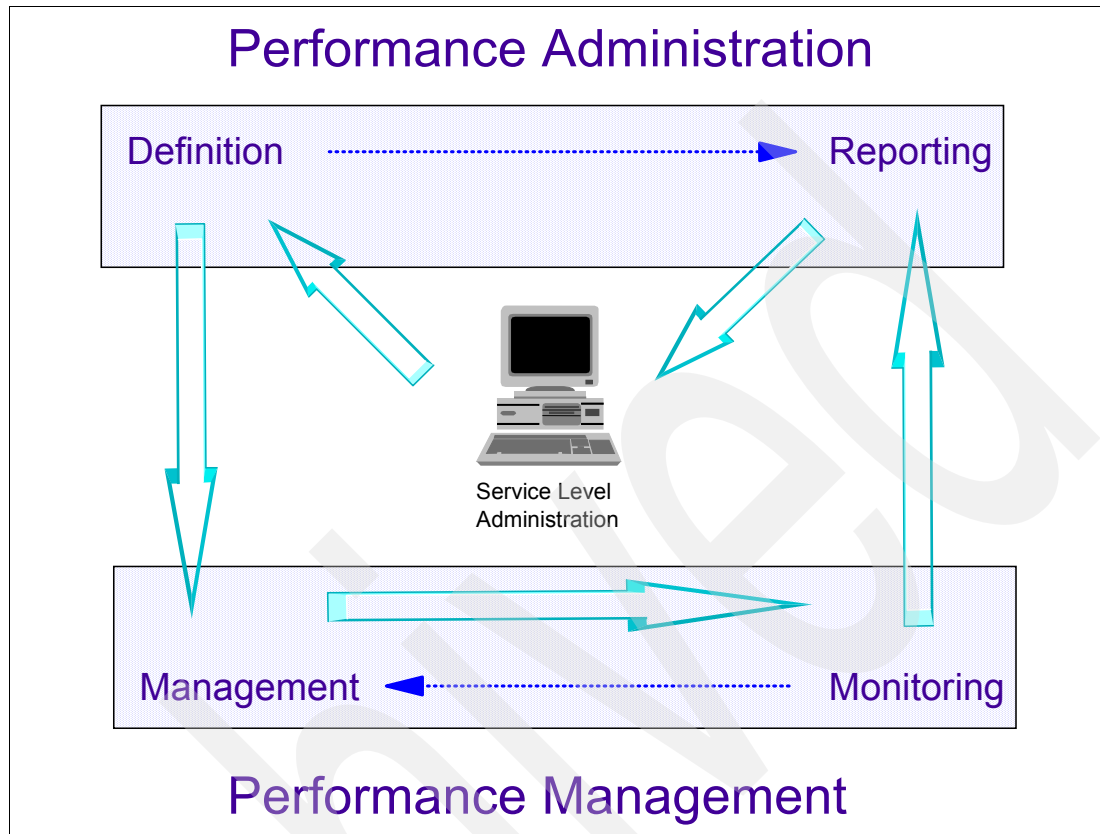


Figure 2-2 Performance analysis overview

Performance analysis

Performance analysis is the set of techniques used to enforce, in your IT systems, the performance goals as defined in the SLA. Perception of system performance can often be somewhat subjective and inaccurate. Service level agreements help to make performance analysis more objective and more effective for business needs.

Performance analysis is composed of two tasks:

- ▶ Performance administration

Performance administration is the process of defining and adjusting performance goals. Workload management introduces the role of the service level administrator. The service level administrator is responsible for defining the installation's performance goals based on business needs and current performance. This explicit definition of workloads and performance goals is called a service definition.

Some installations might already have this kind of information in an SLA. The service definition applies to all types of work including CICS, IMS, TSO/E, z/OS UNIX System Services, JES, APPC/MVS, LSFM, DDF, DB2, SOM, Internet Connection Server (also referred to as IWEB) and others. You can specify goals for all MVS-managed work, whether it is online transactions or batch jobs. The goals defined in the service definition apply to all work in the sysplex.

Because the service definition terminology of WLM is similar to the terminology found in an SLA, the service level administrator can communicate with the installation user community, with upper level management, and with MVS using the same terminology.

When the service level requirements change, the service level administrator can adjust the corresponding workload management terms, without having to convert them into low-level MVS parameters.

- ▶ Performance management

Performance management is the process that workload management uses to decide how to match resources to work according to performance goals. Workload management algorithms use the service definition information and internal monitoring feedback to check how well they are doing in meeting the goals. The algorithms periodically adjust the allocation of resource as the workload level changes.

For each system, workload management handles the system resources. Workload management coordinates and shares performance information across the sysplex. How well it manages one system is based on how well the other systems are also doing in meeting the goals. If there is contention for resources, workload management makes the appropriate trade-offs based on the importance of the work and how well the goals are being met.

Workload management can dynamically start and stop server address spaces to process work from application environments. Workload management starts and stops server address spaces in a single system or across the sysplex to meet the work's goals.

2.3 Performance management heuristics

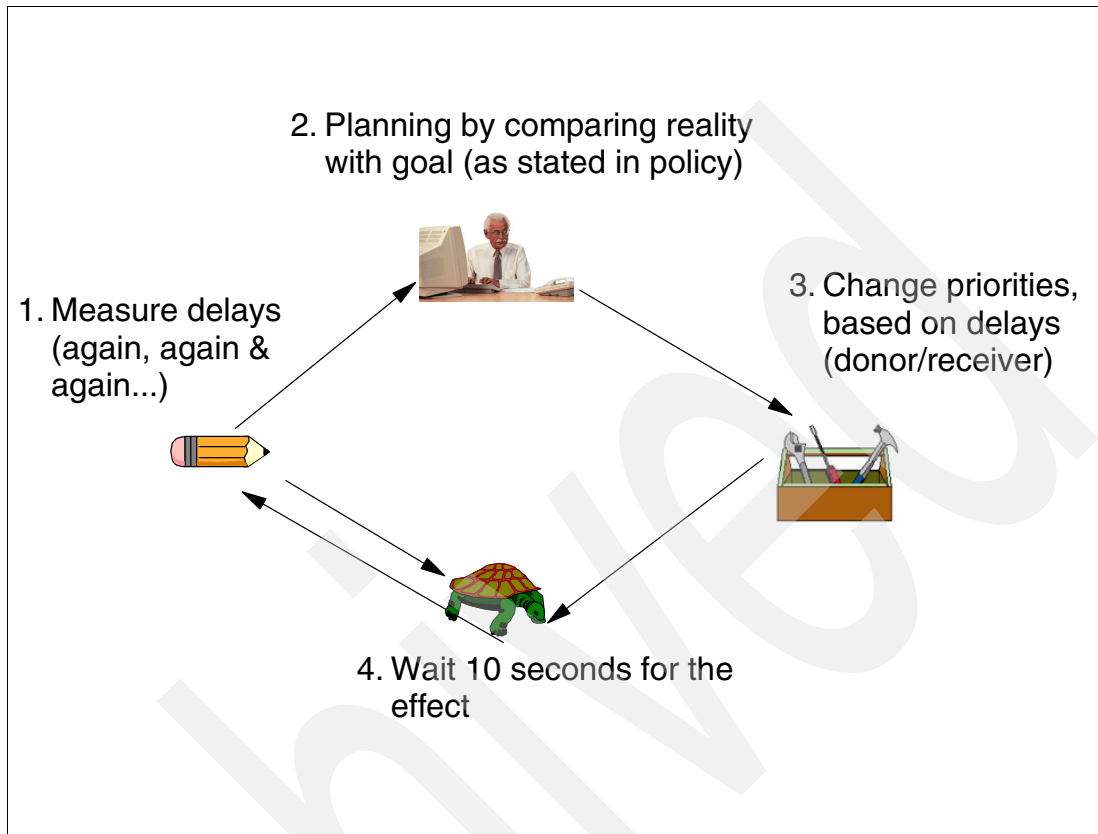


Figure 2-3 Performance management heuristics

Performance management heuristics

Performance management involves a heuristic, ongoing cycle of measuring, planning, waiting for results, and change. It is important to wait between the change and measurement cycles, because a certain amount of time is needed for the system to assimilate the changes. In this context, *heuristic* means reaching a “moving target” state. The major resource performance management tasks are executed by the Workload Manager (WLM) code, based on installation-defined goals. Figure 2-3 on page 57 illustrates the WLM heuristic task.

WLM provides the capability to collect performance data (including delays) every .25 seconds. Planning and changing is performed every 10 seconds (based on 40 samples). WLM performance captured data is available to reporting and monitoring products, so that they can use the same terminology.

Solving performance management problems

Generally speaking, there are three main approaches to solving performance management problems (that is, conflicts between performance reality and the goals stated in the SLA):

- ▶ Buy

The installation can simply purchase more resources.

- ▶ Tune

You can tune your system for more effective and efficient use of resources. In WLM, such function is executed by the WLM Resource Adjustment routine.

► Steal

The installation, or the WLM Policy Adjustment routine, can “steal” resources. That is, you can take resources from a less-critical transaction by modifying priorities in the queues (done by the installation or by the WLM Policy Adjustment routine).

The goal of performance management is to make the best use of current resources to meet current objectives, without investing excessive effort in tuning activities.

If none of these options are technically or financially possible, you need to change the target expectations. You may have experienced the situation where you complete an extensive performance analysis only to conclude that no further tuning or stealing of resources can be done. One goal of this book is to help you determine whether you have reached this point.

It can sometimes be challenging to reconcile the SLA with requirements such as security, availability, integrity, compatibility, and low cost.

2.4 What to measure

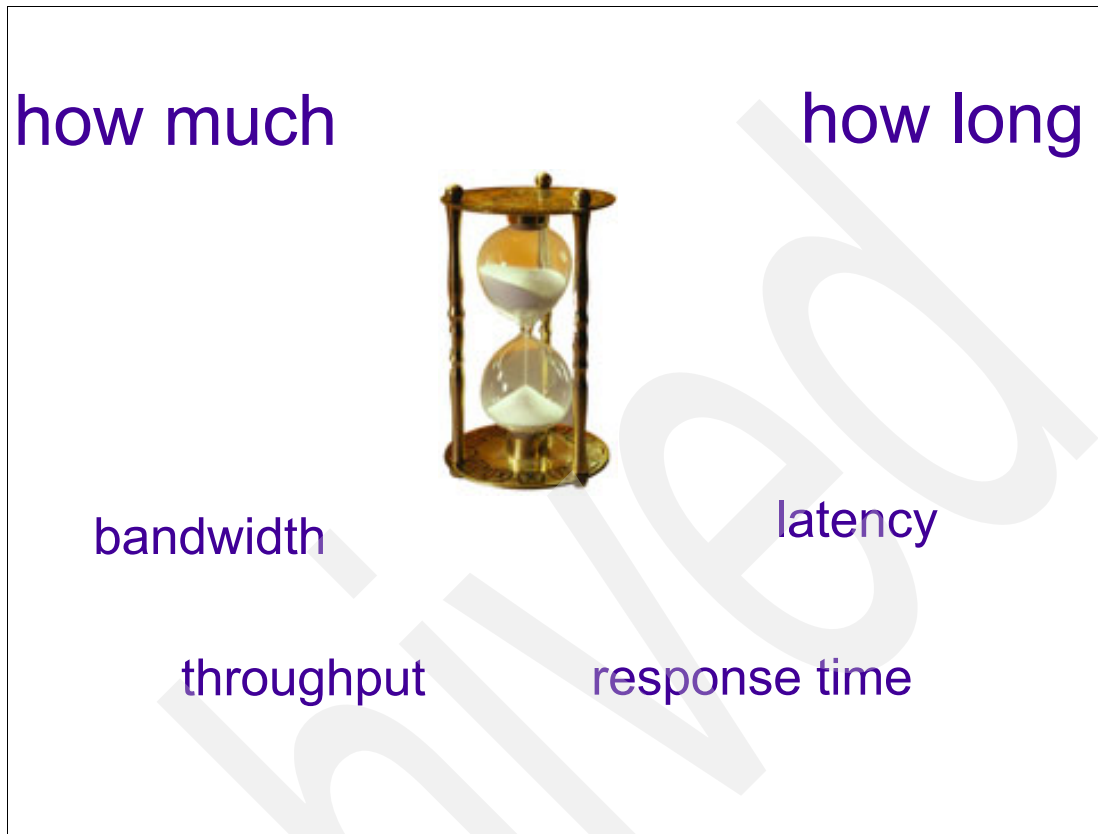


Figure 2-4 What to measure

What to measure

Figure 2-4 on page 59 shows the most important metrics to measure for evaluating system performance.

The leftmost side of the figure lists metrics that are related to production (quantity) aspects of the service: how many jobs per hour, how many kilobytes per second, how many I/Os per second. The rightmost side of the figure lists metrics related to response (quality), such as response time and queue time.

Usually in a system, when the quantity of produced service is high (which is desirable), the quality as response time is also high (which is undesirable) and vice versa.

The metrics and their meanings are explained here:

Bandwidth	This refers to the amount of data that can be transmitted in a fixed amount of time. For digital devices, the bandwidth is usually expressed in bits per second (bps) or bytes per second.
Throughput	This refers to the amount of data transferred from one place to another or processed in a specified amount of time. Data transfer rates for disk drives and networks are measured in terms of throughput. Typically, throughputs are measured in kbps, Mbps and Gbps.
Latency	This refers, in general, to the period of time that one component in a system is “spinning its wheels” waiting for another component. Latency,

therefore, is wasted time. For example, in accessing data on a disk, latency is defined as the time it takes to position the proper sector under the read/write head.

Latency also refers to the time interval between the instant at which an instruction control unit initiates a call for data and the instant at which the actual transfer of the data starts.

Response time

Installations today process different types of work with different response times. Every installation wants to make the best use of its resources, maintain the highest possible throughput, and achieve the best possible system responsiveness.

Response time refers to the amount of time a requester must wait before a request can be granted. It is the elapsed time between the end of an inquiry or demand on a computer system and the beginning of the response, for example, the length of time between an indication of the end of an inquiry and the display of the first character of the response at a user terminal.

For response time monitoring, this term refers to the time from the activation of a transaction until a response is received, according to the response time definition coded in the performance class.

2.5 General performance management metrics

- Average transaction response time
- External throughput rate
- Resource utilization
- Saturation design point

Figure 2-5 Performance management metrics

General performance management metrics

Performance management metrics exist that can be used to verify system behavior towards the SLA, and performance in general. In this context, “system” refers to resources such as PU processors (including CPUs, ICFs, IFLs, zAAPs and SAPs) and channels.

Average transaction response time

The response time of a transaction is defined as the elapsed time between the stop time and the start time, or, in certain cases, the arrival time.

External throughput rate

Performance is measured in a term called the External Throughput Rate (ETR) and it is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user’s job stream, the I/O configuration, the storage configuration, and the workload processed.

Resource utilization

You may also want to consider the concept of Saturation Design Point (SDP) as a complement of the resource utilization in your performance management activity. A resource is said to be critical to performance when it becomes overused or when its utilization is disproportionate to that of other components. For instance, you might consider a disk to be critical or overused when it has a utilization of 70% and all other disks on the system have

30% utilization. Although 70% does not indicate that the disk is severely overused, you can improve performance by rearranging data to balance I/O requests across the entire set of disks.

How you measure resource utilization depends on the tools that your operating system provides for reporting system activity and resource utilization. After you identify a resource that seems overused, you can use database server performance-monitoring utilities to gather data and make inferences about the database activities that might account for the load on that component. You can adjust your database server configuration or your operating system to reduce those database activities or spread them among other components. In some cases, you might need to provide additional hardware resources to resolve a performance bottleneck.

Saturation design point

Saturation of any particular hardware resource prevents you from isolating the requirements of different features and making accurate projections. This is true because saturation limits performance and also limits the utilization of other nonsaturated resources. Because of this, tests need to be carefully scaled to avoid saturation even as they provide a reasonable level of utilization of all hardware resources. As long as any one resource is not saturated, there is always the potential for a higher level of throughput.

2.6 Average transaction response time

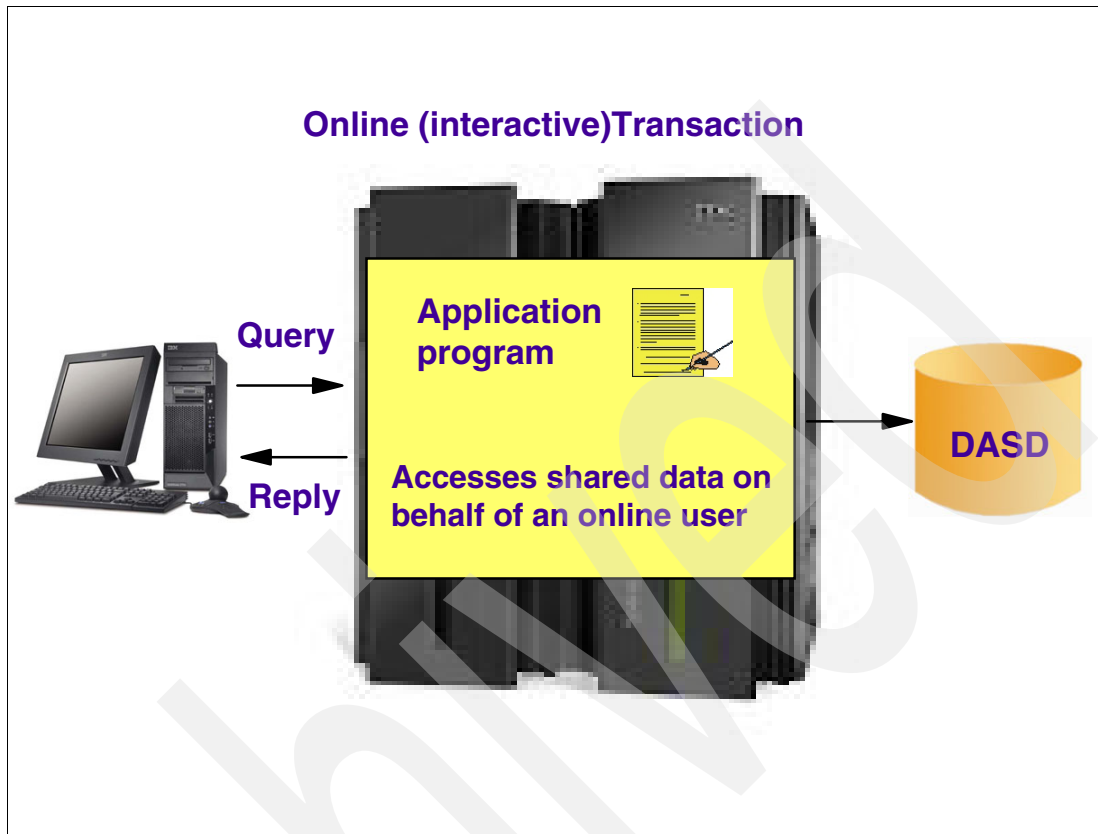


Figure 2-6 Transaction life cycle

Average transaction response time

A *transaction* is a business unit of work implemented to solve a problem for the business. It is produced by the interaction of a user (online or not) and the system. The transaction can be a specific set of input data that triggers execution of a specific process or job, or it can be a message destined for an application program. There are also infrastructure transactions executed to support the business transactions.

Response time measurement is especially important in a client/server environment and is ideally done on a transaction basis. The problem is that a transaction is an elusive concept. Between client and server, transactions can range from causing a single network transmission with no response, to involving a large number of transmissions.

Therefore, transaction time consists of two parts:

- ▶ User think time
- ▶ Response time

User think time

This refers to the time that a user waits between successive actions. It is actually the idle time during which users think about their next action between the first request and another request.

Response time

Response time in an online environment refers to the length of time that elapses between when a user presses Enter and when a window displays the required full data.

Response time for a batch job transaction refers to the time between the JOB submit and the final execution of the last job step. The time used for sysout printing and purging is not included.

Response time components

Response time consists of:

- ▶ Host response time
- ▶ Network time
- ▶ Other servers' time

The following list shows major steps in a client/server application. Notice that each step lists resources that are required by the task. These resources, therefore, will influence the response time component if the resources are scarce.

1. Client application processes user input (CPU, disk)
2. Client machine enqueues network request (CPU, adapter, network)
3. Request is transferred over network (network capacity and speed)
4. Server enqueues request (CPU, adapter)
5. Server application processes request (CPU, disk, possibly access to other servers)
6. Server machine enqueues response (CPU, adapter, network)
7. Response is transferred over network (network capacity and speed)
8. Client application processes response (CPU, disk, possibly access to other servers)
9. Client application sends response to user (CPU, terminal network).

ATM cash withdrawal example

The simple act of withdrawing cash from an automated teller machine (ATM) is much more complicated than it appears. You begin by inserting your identification card and entering a personal identification number (PIN). Your identity is verified online when a computer in the network compares the information you entered to a database of customers belonging to that financial institution. Internal electronic messages are created to access the specific checking or savings account where the money is held. Then, the account balance is verified and approved. Finally, a message is sent back to the ATM to disperse the funds or refuse the transaction.

The withdrawal transaction triggers secondary transactions to update the appropriate checking or savings accounts; this is usually done in real-time. By the time the money is dispensed from the machine, the account balance will reflect the withdrawal. It becomes more complex if you make an out-of-territory withdrawal, for example, if you use Bank 1's ATM to withdraw money from your account at Bank 2. The peer bank's database must be accessed and the account status verified.

Note: All of this occurs as the customer waits at the machine. The network and mainframe computers involved must be fast to keep the transaction response time "reasonable" from the customer's point of view. The successful completion of the transaction depends on, among other things, both banks using compatible network technology to exchange information.

2.7 Transaction response time components

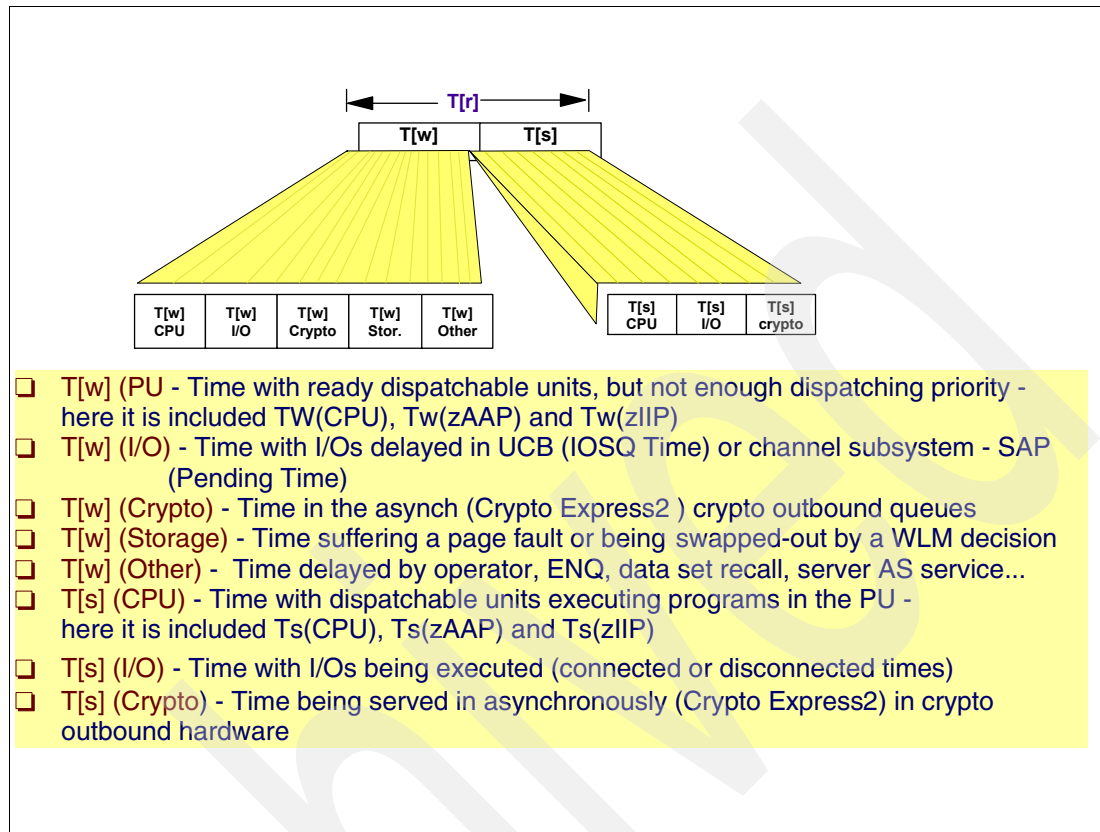


Figure 2-7 RMF terminology for transaction resources

Transaction response time components

Response time is made up of service time (the time actual work is done) and waiting time (the time waiting for resource) from an RMF perspective, as follows:

$$T[r] = T[s] + T[w]$$

Where:

- ▶ T[r] is Response Time
- ▶ T[s] is Service Time
- ▶ T[w] is Waiting Time (that is Queue time or Delay in RMF Monitor III)

Similarly, total transaction service and wait times are made up of the individual resource service and wait times, as follows:

$$T[s] = T[s] \text{ (CPU)} + T[s] \text{ (I/O)} + T[s] \text{ (TP)} + T[s] \text{ (Other)}$$

$$T[w] = T[w] \text{ (CPU)} + T[w] \text{ (I/O)} + T[w] \text{ (TP)} + T[w] \text{ (Storage)} + T[w] \text{ (Other)}$$

The RMF Monitor I Workload Activity report shows some of this data for a group of transactions (service or report class). This applies for TSO and batch work (and potentially CICS, with interval recording).

Transaction response time terminology

The response time of a transaction is defined as the elapsed time between the stop time and the start time, or, in some cases, the arrival time. The key to finding and fixing any transaction I/O-related performance problems is DASD response time, that is, the length of time it takes to complete an I/O operation. Response time can have a dramatic effect on performance, particularly with online and interactive subsystems.

DASD response time is the elapsed time from the DIAGNOSE instruction at the start subchannel (SSCH) instruction to the completion of the data transfer, which is indicated by a channel end/device end (CE/DE) interrupt. It includes any queue time plus the actual I/O operation.

Service time is the elapsed time from the successful SSCH instruction to the data transfer completion. It includes seek time, any rotational delays, and data transfer time. Service time plus queue time equals response time.

Queue wait time

Queue wait time is the internal queuing of the I/O operations that are waiting for a previous I/O to the device to complete. Queue time represents the time spent waiting on a device. Delays in the other service components can cause the queue component to increase, or the queue time may be a function of skewed arrival of I/Os from a particular application.

Pending time

Pending time is the time from the start of the I/O until the DASD receives it. The pending time indicates the channel path usage. A high pending time indicates that the channel or logical control unit is busy. Pending time can be caused by busy channels and controllers or device busy from another system.

Transaction response time is the best way to characterize the performance of a transaction, because it is directly connected to business needs and it is easily understood by people. The response time that is measured by z/OS and reported by the Resource Measurement Facility™ (RMF) is the host response time as pictured in AVG field in Figure 2-9.

For more detailed information about RMF, see 3.2, “RMF Performance Management panel” on page 178.

Disconnect time

Disconnect time includes:

- ▶ The time for a seek operation.
- ▶ Latency always assumed to be half a revolution of the device.
- ▶ Rotational position sensing (RPS) reconnect delay, which is the time for the set sector operation to reconnect to the channel path. This time depends on internal path busy, control unit busy, and channel path busy.

If any element in the path is busy, a delay of one revolution of the device is experienced.

When a device cannot reconnect to the host to transfer data because all paths are busy, it must wait for another revolution. Using cache control units reduces or eliminates disconnect time. Disconnect time is used as a measurement of cache effectiveness.

Connect time

Connect time is the time actually spent transferring data between the channel and DASD or channel and cache. Connect time can also include the time a search operation is occurring

between the channel and the DASD or the channel and the cache (usually done to search a directory to find the location of a program module so it can be loaded into storage).

A high connect time indicates that you are using large blocks to transfer the data. This can be a problem if you mix small blocks and large blocks. The small blocks may have to wait for the larger ones to complete, thus causing a delay.

Archived

2.9 External throughput rate (ETR)

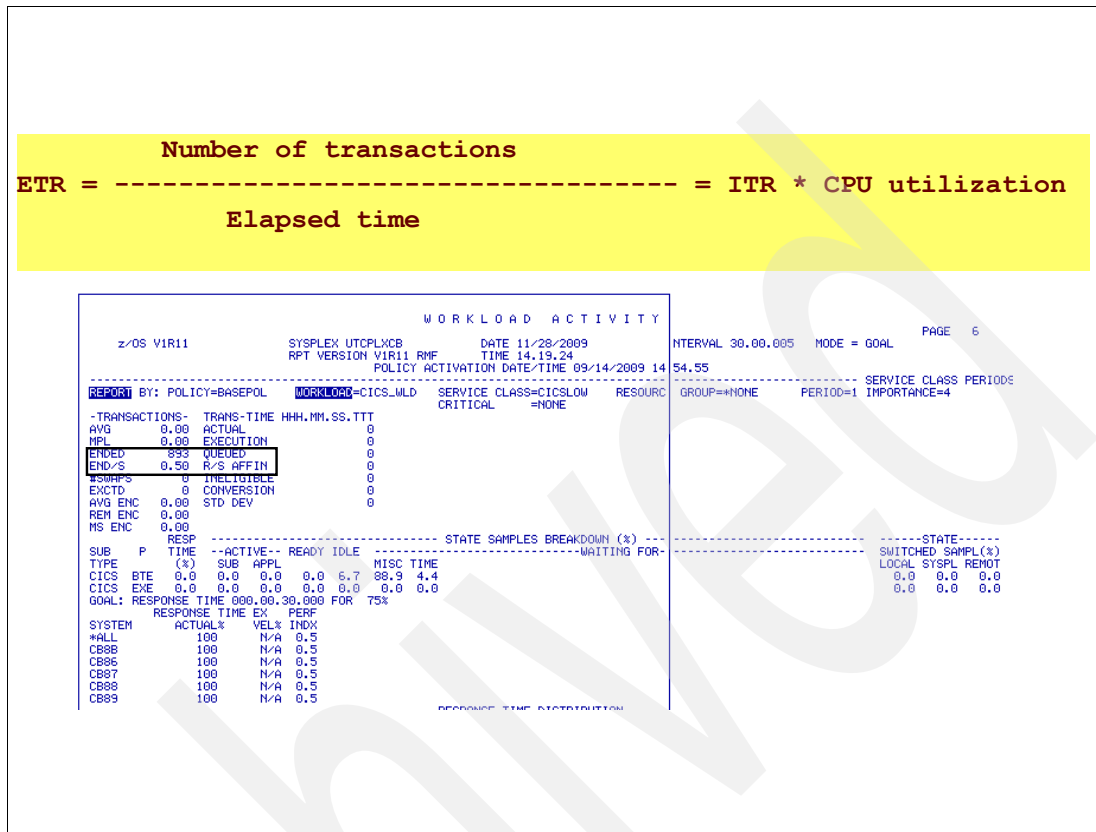


Figure 2-9 External throughput rate (ETR)

External throughput rate (ETR)

External throughput rate (ETR) is a measurement of the number of ended transactions per elapsed time, as shown in Figure 2-9 on page 69. ETR is also called the *transaction rate*. It is associated with a system's throughput.

Note: Any bottleneck, internal or external to the system, will effect the ETR. Examples include I/O constraints, tape mount delay, and paging delay. Thus it is more difficult to get a repeatable measure than with ITR.

RMF Workload Activity report

In the RMF Workload Activity report in Figure 2-9 on page 69, the interval set was 30 minutes (1800 seconds), where 893 transactions from service class CICSLOW ended, causing an ETR of 0.50 (field END/S in the report).

There is also a formula that relates the $T[r]$ with ETR, derived by Little's Law (see 2.64, "Little's Law" on page 164 for more information about this topic):

$$ETR = N / (T[t] + T[r])$$

Where:

- ▶ N is the average number of users sending transactions (logged on)
- ▶ $T[t]$ the average thinking time of these users
- ▶ $T[r]$ is the average response time

Any bottleneck, internal or external to the system, affects the ETR. Examples include I/O constraints, tape mount delay, paging delays, operator delays and other delays.

Considerations regarding this formula

The variables that affect the ETR most are N and Tt, because of their usual high numeric values. Therefore, avoid service level agreements that specify an ETR because the only variable that an IT department can directly control is Tr.

However, experience shows that when Tr is in a sub-second value, the value of Tt drops dramatically. The target is to reduce the Tt (linked to human productivity) when the value is below one second, and consequently to increase the ETR. The graphic interface (GUI) of a personal computer can help to generate a low Tt; however, other PC tasks can increase such values.

In the mentioned formula, derived by Little's Law, N is the average number of users sending transactions (logged on), and Tt is the average think time of these users.

$$ETR = N / (Tt + Tr)$$

Note the following considerations regarding this formula:

- ▶ The variables that more intensively affect the ETR are N and Tt due to their usual numeric values. Therefore, never accept an SLA specifying ETR, because the only variable that the IS department can directly control is Tr.
- ▶ However, experience shows that when Tr is in a sub-second value, the value of Tt drops dramatically. This fact has to do with the human behavior in front of a terminal; if the machine responds fast (in a sub-second value also respond fast).

Transaction response time is the best way to characterize the performance of a transaction. Here, the target is to reduce its value and consequently to increase the ETR figures (when the value is less than one second). Remember that in RMF reports, the Ts(TP) and Tw(TP) are not included in the response time pictured.

2.10 Resource utilization

- ❑ If your system is showing signs of CPU constraint
 - See the CPU Activity Report
 - Check the Partition Data report to be sure that your LP is correctly configured
 - You would do this to allocate more CPU resource to a particular LP

CPU ACTIVITY												
z/OS VIR11				SYSTEM ID S59				DATE 11/28/2009		INTERVAL 14.59.997		
RPT VERSION VIR11				RMF				TIME 16.45.00		CYCLE 1.000 SECONDS		
E26 SEQUENCE CODE 00000000005C34F				HUPERDISPATCH=YES				PAGE 1				
CPU	2097	MODEL	720	H/W	MODEL	E26	SEQUENCE	CODE	00000000005C34F	HUPERDISPATCH=YES		
---CPU---												
NUM	TYPE	ONLINE	LPAR	BUSY	MVS	BUSY	PARKED	SHARE	%	RATE	%	VIA TPI
0	CP	100.00	99.96	100.0	0.00	100.0	0.00	100.0	HIGH	95.31	0.00	0.00
1	CP	100.00	99.58	100.0	0.00	100.0	0.00	100.0	HIGH	0.00	0.00	0.00
2	CP	100.00	99.58	99.97	0.00	100.0	0.00	100.0	HIGH	0.00	0.00	0.00
3	CP	100.00	99.58	99.97	0.00	100.0	0.00	100.0	HIGH	0.00	0.00	0.00
4	CP	100.00	99.58	99.98	0.00	100.0	0.00	100.0	HIGH	0.00	0.00	0.00
5	CP	100.00	78.17	100.0	0.00	70.3	0.00	70.3	MED	0.00	0.00	0.00
6	CP	100.00	78.10	100.0	0.00	70.3	0.00	70.3	MED	0.00	0.00	0.00
7	CP	100.00	0.01	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
8	CP	100.00	0.01	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
9	CP	100.00	0.01	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
A	CP	100.00	0.01	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
B	CP	100.00	0.01	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
C	CP	100.00	0.01	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
D	CP	100.00	0.01	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
E	CP	100.00	0.01	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
F	CP	100.00	0.01	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
10	CP	100.00	0.01	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
11	CP	100.00	0.00	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
12	CP	100.00	0.00	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
13	CP	100.00	0.00	0.00	100.00	0.0	LOW	0.00	0.00	0.00	0.00	0.00
TOTAL/AVERAGE			32.76	34.99		640.6		95.31		0.03		

Figure 2-10 Formulas and RMF CPU activity report - extract

Resource utilization (U%)

Monitor I and Monitor III provide long-term data collection about system workload and resource utilization, and cover all hardware and software components of your system: processor, I/O device and storage activities and utilization, underscores consumption, activity, and performance of groups of address spaces.

Resource utilization measures how much a resource delivers service in a timely basis. The general formula is:

$$U\% = \text{Busy_Time} \times 100 / \text{Elapsed_Time}$$

We may derive (for just one server):

$$\text{Busy_Time} = T[s] \times \text{Transactions}$$

Replacing Busy_Time in the first formula:

$$U\% = T[s] \times \# \text{Transactions} \times \text{Elapsed_Time} / 100$$

$$U\% = T[s] \times \text{ETR} \times 100$$

This formula shows that the U increases when T[s], ETRs, or both, rise.

For example, if a bank branch office with only one teller working experiences an average arrival rate of two clients/minute with a service time (Ts) of 0.25 minute, we can say that the teller average utilization is $U\% = 0.25 \times 2 \times 100 = 50\%$.

Note the following points:

- ▶ The product $T[s] \times ETR$ is also called *Traffic*.
- ▶ The product $T\{r\} \times ETR$ is called *Intensity*.

LPAR busy time percentage

Figure 2-10 shows the RMF CPU Activity report portraying the average logical CPU utilization for reported logical partitions. For example, this logical partition has 19 logical CPs with an average utilization of 32.76%.

There is also a Markov's equation that relates T_w and $U\%$. Refer to 2.65, "Markov's Equation" on page 166 to see the curve representing the following equation:

$$T[w] = T[s] * U / 1-U$$

The graphic in Figure 2-67 shows that for a $T[s]$ constant, for $U\%$ greater than 35%, $T[w]$ increases drastically.

2.11 Physical PU utilization example

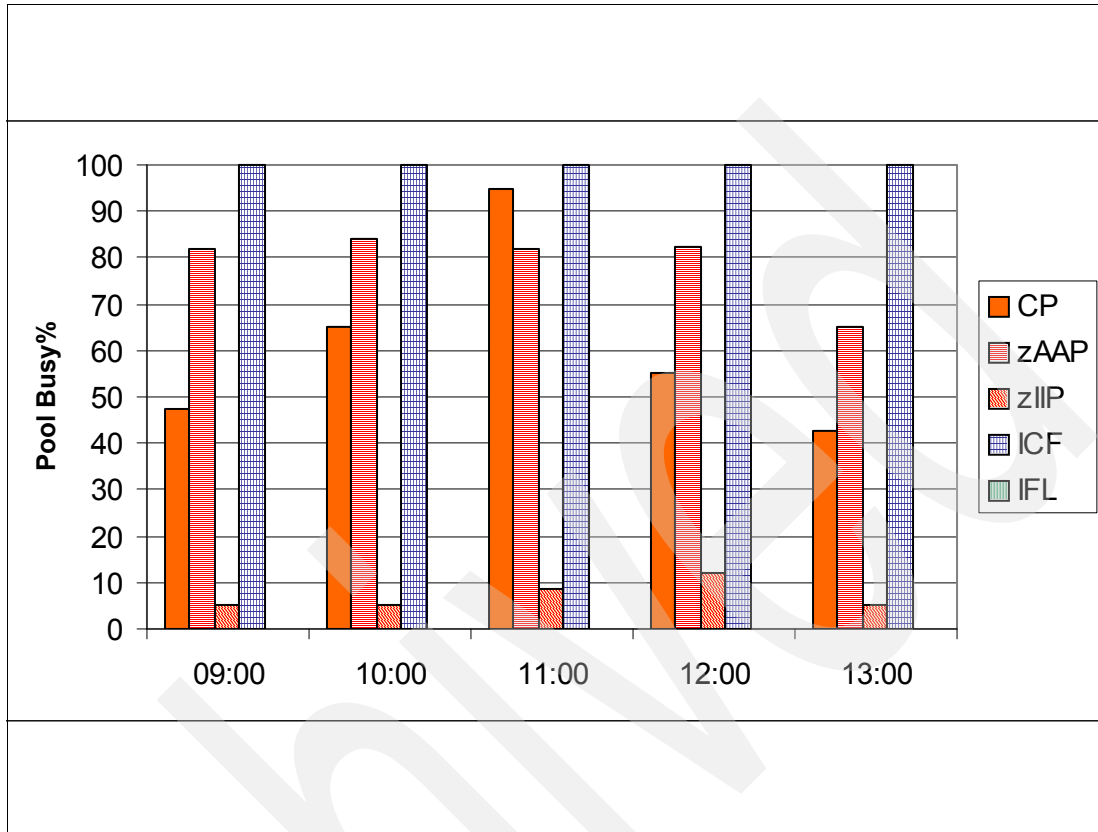


Figure 2-11 Physical PU Utilization example

Physical PU Utilization example

Figure 2-11 on page 73 shows a graphic produced by the RMF Spreadsheet Reporter portraying the utilization of physical PUs (CP, zAAP, zIIP, ICF, and IFL) over a period of time in one server (that is the processor CPU or server). Each PU is described briefly here:

- ▶ CPU is a PU that can run any mainframe operating system.
- ▶ zAAP and zIIP are PUs that can run only in a z/OS operating system.
- ▶ ICF is a PU that can run only the Coupling Facility (CFCC) operating system.
- ▶ IFL is a PU that can run Linux® on System z® native, or the pair of z/VM® with Linux on System z.

Because the CFCC is in an eternal loop, ICF utilization is always 100%.

RMF Spreadsheet Reporter

The RMF Spreadsheet Reporter is the powerful workstation solution for graphical presentation of RMF Postprocessor data. Use it to convert your RMF data to spreadsheet format and generate representative charts for all performance-relevant areas. The RMF Spreadsheet Reporter offers the following features:

- ▶ Ease of use - manage the related resources by means of an Explorer-like GUI
- ▶ Fast path to graphical presentation - prepare the SMF data in a single step
- ▶ Batch mode - generate the input files for the spreadsheets without any GUI interaction

2.12 Channel utilization example

```

RMF - CHANNEL Channel Path Activity
Command ==>
CPU= 37/ 35 UIC=2540 PR= 0
System= CB88 Total
Line 1 of 69
Scroll ==> HALF
08:01:56 Channel Utilization(%) Read(B/s) Write(B/s) FICON OPS zHPF OPS
ID No G Type S Part Tot Bus Part Tot Part Tot Rate Act Rate Act
4 *CNCSM 0.1 0.5
4 *FC_SM 0.0 0.0 0.0 0 0 0 0
12 OSD Y 0.0 0.0 0.0 2K 19K 0 0
14 OSD Y 0.0 0.0 0.0 5K 478K 458K 461K
16 OSD Y 0.4 1.3 0.0 493K 5M 3M 5M
20 CTC_S Y 0.0 0.0
27 CNC_S Y 0.0 0.0
2B CNC_S Y 0.9 3.7
2C CNC_S Y 0.2 0.6
30 5 FC_S Y 0.0 31.8 8.6 301 50M 71 227K 177 1 0 0
31 5 FC_SM Y 0.0 31.6 7.9 239 46M 220 235K 177 1 0 0
37 4 FC_S Y 0.0 0.1 0.0 0 85K 0 26K 6 1 0 0
38 4 FC_S Y 0.0 0.1 0.0 0 69K 0 36K 7 1 0 0
39 4 FC_S Y 0.0 0.0 0.0 390 11K 0 15K 2 1 0 0
3A 4 FC_S Y 0.0 0.0 0.0 0 13K 0 16K 3 1 0 0
3E 4 FC_S Y 0.0 0.0 0.0 0 1K 0 568 0 1 0 0
7C CNCSM Y 0.4 1.8
7D CNCSM Y 0.0 0.1
81 3 FC_S Y 2.3 20.2 5.3 3M 30M 671K 2M 998 3 214 1
82 5 FC_S Y 0.1 0.9 0.3 147K 2M 282 89K 30 1 48 1
83 5 FC_S Y 0.1 0.9 0.3 162K 2M 291 86K 30 1 48 1
84 4 FC_S Y 0.0 0.0 0.0 56 223 0 0 1 1 0 0
85 3 FC_S Y 0.3 13.4 1.5 45K 4M 43K 2M 842 1 356 1
8C 3 FC_S Y 0.9 10.9 1.6 585K 7M 80K 1M 718 2 0 0
A6 5 FC_SM Y 0.0 0.0 0.0 0 0 0 0 0 0 0 0
B6 5 FC_SM Y 0.0 0.0 0.0 0 0 0 0 0 0 0 0
E0 IQD Y 0 346K
E1 IQD Y 0 0
E2 IQD Y 0 0
E3 IQD Y 0 0

```

Figure 2-12 Channel utilization example

Channel utilization example

Channels are special processors involved in dialogs with I/O controllers during the execution of an I/O operation. The Channel Path Activity report (CHANNEL) gives you information about channel path activity for all channel paths in the system. The report contains data for every channel path that is online during data gathering.

You can use channel path activity information together with I/O device activity and I/O queuing activity information to identify performance bottlenecks associated with channel paths.

Note: To determine which logical control unit is using the channel, look in the I/O Queuing Activity report. From there you can proceed to check device response times. For example, if a channel path to a device shows excessive use, you can define additional paths to the device or introduce a different job mix to produce better performance.

Figure 2-12 on page 74 shows an RMF Channel Activity report displaying the utilization of all channels located in a CEC. For example, the channel with CHPID 30 has an average utilization of 31.8%.

2.13 Saturation design point (SDP)

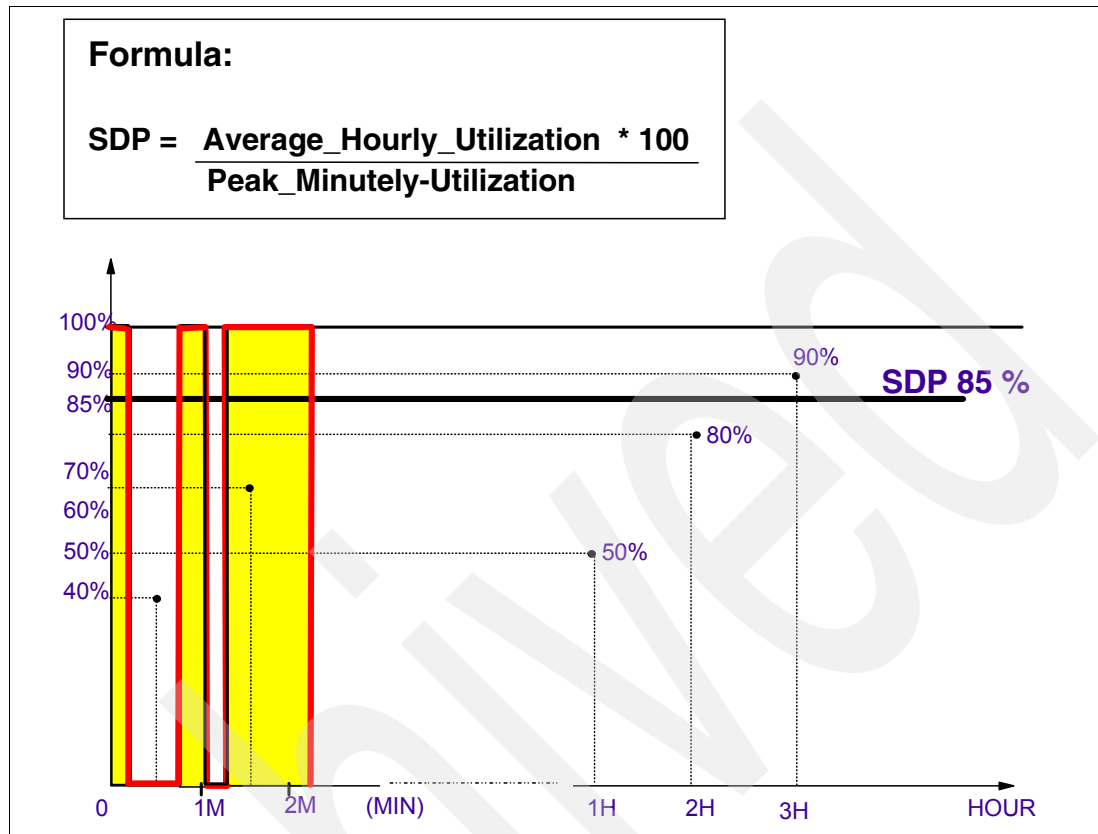


Figure 2-13 Utilization curve with SDP

Saturation design point (SDP)

When considering capacity planning, it is important to consider the service levels of the work eligible to execute. It may be typical to assume a utilization of 90% or higher in capacity planning; this is referred to as the saturation design point (SDP). Utilizations of 90% often can result in wait-to-using averages of 10 to 20, meaning the work will wait for the processor 10 to 20 times as long as its service time at the processor. This is often acceptable because there is work with a relaxed response time objective to utilize the machine after the work with stringent response objectives has been satisfied. The most important applications are not expected to experience those high delays. WLM has a designation of “discretionary” work in the WLM service definition to make clear the lack of a service level for some work. One consideration for the most important work is to use an SDP of your choice.

Planning for SDP

A CPU is either in busy state or in wait state. In other words, either it is instantaneously 100% busy or 0% busy. Figure 2-13 on page 75 illustrates this point along a certain amount of time scaled in minutes. The shaded area indicates that the CPU is busy. Performance monitors group these states in a specific amount of time, for example, one minute. The leftmost side of the graphic shows a minute time scale. For example in the first minute, the average utilization is calculated as 40%. In the second minute, average utilization is 65%.

Usually these values in minutes are grouped, for example, in hours. An hourly average is calculated and presented for human analysis (50% in the first hour, 80% in the second hour,

and 90% in the third hour, shown in the rightmost side of the graphic). As you decrease the granularity (from one minute to one hour), you save space to store the numbers (which is desirable) but you can lose key details (which is undesirable). The question then becomes, if the hourly utilization is 75%, how often was 100% reached in the *per minute* average within this hour?

SDP concept

To answer this question, the concept of saturation design point was developed. SDP refers to the maximum average utilization in a larger interval (one hour, for example), where the installation is sure that in the smaller interval (one minute, for example), the average utilization was never 100%. If the current average is equal or greater than the SDP, then the 100% busy was reached in the minor interval. The formula for the hour/minute relationship is as follows:

$$\text{SDP\%} = \text{Average Hourly Utilization} * 100 / \text{Peak Minutely Utilization}$$

Imagine that, in some hour interval, the Average Hourly Utilization was 80%, and in that hour interval, the highest minute average peak was 92%. Then:

$$\text{SDP\%} = 80 * 100 / 92 = 87\%$$

Observing the formula, it is clear that when the SDP is equal to the Average Hourly Utilization, then at least in one minute the 100% minute average is reached. In Figure 2-13, the SDP reaches 85%. When the hour Utilization is 90% (as in the third hour), there is a guarantee that in some minute average within that hour, the 100% mark was reached.

If a client knows its SDP using theoretical (usually 70%) or experimental approaches, it can apply this information:

- ▶ In performance management, by avoiding having the hour average exceed the SDP.
- ▶ In capacity planning, by sizing processors to avoid exceeding the SDP threshold, at least as a capacity starting point.

2.14 Processor performance metrics

- ❑ CPU Time
 - Cycle Time
 - Cycles per Average Instruction (CPAI)
 - Path length
- ❑ MIPS and consumed MIPS
- ❑ MFLOPS
- ❑ CP Service Units
- ❑ Millions of Service Units per Hour (MSU/h)
- ❑ Internal Throughput Rate (ITR)
- ❑ Relative Processor Power (RPP)

Figure 2-14 Processor performance metrics

Processor performance metrics

The following sections introduce metrics used in performance management, software pricing charging, capacity planning, charge back, and competition scenarios to evaluate processor speed capacity:

- ▶ CPU time
- ▶ MIPS and consumed MIPS
- ▶ MFLOPS
- ▶ CP service units - SRM constant - MSU
- ▶ Internal Throughput Rate (ITR)
- ▶ Large Systems Performance Reference (LSPR) - RPP

2.15 CPU time

- CPU time = Path length * (Cycles / Instruction) * Cycle time
 - Path length depends on instruction set and compiler
 - Cycles/instruction depend on design (for example, microcoding of instructions)
 - Cycle time depends on technology (for example, CPU model)

WORKLOAD ACTIVITY																																	
z/OS V1R11			SYSPLEX SVPLEX3			DATE 11/28/2009			INTERVAL 14.59.995			MODE = GOAL			PAGE 1																		
RPT VERSION V1R11 RMF			TIME 12.00.00			POLICY ACTIVATION DATE/TIME 09/14/2009 10			12.11																								

REPORT BY: POLICY=BASEPOL			WORKLOAD=STC_MLD			SERVICE CLASS=STCLOW			RESOURC			GROUP=#NONE			PERIOD=1			SERVICE CLASS PERIODS															

			CRITICAL			=NONE									IMPORTANCE=3																		

-TRANSACTIONS-		TRANS-TIME		HHH.MM.SS.TTT		--DASD		I/O--		--SERVICE--		SERVICE T		ME		---APPL %---		---PROMOTED---		---STORAGE---													
AVG		153.37		ACTUAL		3.02.885		SSCHRT		56.9		IOC		3964		CPU		805.97		CP		92.24		BLK		1.489		AVG		1195.43			
MPL		152.35		EXECUTION		3.02.391		RESP		15.1		CPU		15184K		SRB		13.50		AAPCP		0.00		ENQ		0.046		TOTAL		182122.4			
ENDED		599		QUEUED		494		CONH		1.3		MSD		9		RCT		9.95		IIPCP		0.00		CRM		5.533		SHARED		230.59			
END/S		0.67		R/S AFFIN		0		DISC		0.3		SRB		261005		IIT		0.75		HST		0.00		AAP		0.00		-PAGE-IN RATES-		SINGLE		0.0	
#SWAPS		3391		INELIGIBLE		0		Q+PEND		4.5		TOT		15449K		HST		0.00		AAP		0.00		IIP		0.00		BLOCK		0.0			
EXCTD		0		CONVERSION		5.188		IOSQ		9.0		/SEC		17202		AAP		0.00		IIP		0.00				SHARED		0.0					
AVG ENC		0.00		STD DEV		3.27.429										IIP		0.00						HSP		0.0							
REM ENC		0.00																															
NS ENC		0.00																															
GOAL: EXECUTION VELOCITY		25.0%		VELOCITY MIGRATION:		I/O MGMT		15.6%		INIT		MGMT		15.6%																			
RESPONSE TIME EX		PERF		AVG		--EXEC USING%--																											
SYSTEM		VEL%		INDX		ADRSP		CPU AAP IIP I/O		TOT CPU I/O MPL		EXEC DELA		S %		-USING%--		---		DELAY %		---											
*ALL		--N/A--																															
D0		93.7		0.3		26.5		0.6 0.0 0.0 0.1		0.8 0.0 0.0 0.0		0.0 1.4		21		78		0.0 1.4		0.0													
D2		12.9		1.9		211.1		0.4 0.0 0.0 0.0		2.5 1.8 0.4 0.3		0.0 0.2		29		68		0.0 0.2		0.0													
D4		89.3		0.3		24.8		0.0 0.0 0.0 0.0		0.0 0.0 0.0 0.0		0.0 1.8		21		79		0.0 1.8		0.0													
D6		2.3		11.0		30.8		0.0 0.0 0.0 0.0		0.6 0.0 0.0 0.0		0.0 1.4		17		82		0.0 1.4		0.0													

Figure 2-15 CPU time formula and RMF Workload Activity report

CPU time

The CPU time of a transaction is the processing time consumed to execute all the programs logic associated with such transaction. In z/OS, the program is mapped into a dispatchable unit task (TCB) or service request block (SRB), and it runs in an address space or enclave. The transaction CPU time is measured by z/OS.

The RMF Workload Activity report extracted in Figure 2-15 on page 78 shows several components of the CPU time in a service class period basis. The captured CPU time (in seconds) components are the fields boxed in the figure: CPU, SRB, RCT, IIT, HST, AAP, and IIP.

CPU time here also includes zAAP (AAP) and zIIP (IIP) time, as shown.

CPU time metric

CPU time is the most important metric for evaluating processor speed. The faster the processor, the less CPU time is consumed. However, CPU time is not repetitive because it depends on the following considerations:

- ▶ CPU speed

When changing CPU models, the same program has different CPU times.

- ▶ The set of executed instructions

The same COBOL source code, compiled with different versions of compiler or with the same version but with different optimization parameters, when executed in the same CPU model can present a different CPU time, depending on optimization factors.

- ▶ CPU utilization

Due to the huge population of current running transactions and the complexity of the z10 and z/OS design to support different loads, the CPU time consumed by a program depends on the average CPU utilization. The higher the utilization, the higher the needed CPU time for the same logic.

Note: There are times when additional details of CPU utilization are needed. For capacity planning especially, you may need to know the average CPU time consumed per transaction. This is a simple calculation, dividing application time by the number of transactions.

One of the objectives of performance management is to decrease CPU time.

CPU time

The CPU time value can also be theoretically derived through the formula:

$$\text{CPU time} = \text{Path length} * (\text{Cycles} / \text{Instruction}) * \text{Cycle time}$$

In the following sections, each element of this formula is discussed.

2.16 z10 Cycle Time and Server Time Protocol

- ❑ Each core on the chip runs at a cycle time of 0.227 nanoseconds (4.4 GHz)
- ❑ Number of cycles per average instruction depends greatly on the set of program executed instructions
- ❑ Server Time Protocol (STP) is a server-wide facility that is implemented in the Licensed Internal Code for:
 - System z servers and Coupling Facilities
 - STP presents a single view of time to PR/SM
 - To maintain time synchronization
- ❑ Path lengths

Figure 2-16 z10 cycle time and timer

Cycle time

Cycle time depends on the time to transport the pulse across the circuitry and the switch time; that is, the time to invert the 0s to 1s and vice versa within the Boolean circuits.

The technology selected, applied to a processor, determines the value of the cycle time. The shorter the cycle time, the faster the processor and less CPU time consumed, if the other formula elements are constant. For example, the z10 processors have a cycle time of 0.227ns, corresponding to a frequency of 4.4 GHz.

Thus, cycle time by itself is not a complete indicator of CPU speed, because of the lack of the the two components that compose the CPU time.

Cycle per average instruction

The number of cycles per average instruction depends greatly on the set of program executed instructions. There are instructions that are executed in just one cycle. The z10 can even execute more than one instruction in one cycle (this property is called *superscalar*). On the other hand, there are complex instructions that need many cycles to execute, which consequently biases the cycles per average instruction needed to a higher value.

The lower the cycles per average instruction, the better. The cycles per average instruction depend on the design of the computer; the better the design, the lower the cycles per average instruction and the less CPU time is needed to execute the transaction. Computer design includes techniques such as internal parallelism, CPU cache, and pipeline.

Server Time Protocol facility

Server Time Protocol (STP) is a server-wide facility that is implemented in the Licensed Internal Code of System z servers and Coupling Facilities. STP presents a single view of time to PR/SM™ and provides the capability for multiple servers and Coupling Facilities to maintain time synchronization with each other. Any System z servers or Coupling Facilities can be enabled for STP by installing the STP feature. Each server and CF that are planned to be configured in a coordinated timing network (CTN) must be STP-enabled.

The STP feature is designed to be the supported method for maintaining time synchronization between System z servers and Coupling Facilities. The STP design uses the CTN concept, which is a collection of servers and Coupling Facilities that are time-synchronized to a time value called coordinated server time.

Network Time Protocol (NTP) client support has been added to the STP code on the System z10® and on System z9®. With this functionality the System z10 and the System z9 can be configured to use an NTP server as an external time source (ETS).

This implementation answers the need for a single time source across the heterogeneous platforms in the enterprise, allowing an NTP server to become the single time source for the System z10 and the System z9, and other servers that have NTP clients (UNIX, NT, and so on). NTP can only be used for an STP-only CTN where no server can have an active connection to a Sysplex Timer®.

The time accuracy of an STP-only CTN is improved by adding an NTP server with the pulse per second output signal (PPS) as the ETS device. This type of ETS is available from several vendors that offer network timing solutions. Improved security can be obtained by providing NTP server support on the Hardware Management Console (HMC), because the HMC is normally attached to the private dedicated LAN for System z maintenance and support.

Path length

Path length refers to the number of instructions needed to execute the transaction program that consumed the CPU time shown in the formula in 2.15, “CPU time” on page 78. A shorter path length results in a shorter CPU time.

Path length depends on both of the following:

- ▶ Processor architecture, which defines the set of instructions
- ▶ The quality of the compiler that generates the object code

A well-designed architecture offers a powerful instruction set to implement a program with a short path length. Without further instrumentation, it is not possible to derive theoretically the numeric value of the CPU time for a running program. In the formula referred at 2.15, “CPU time” on page 78, we are only able to access the cycle time.

2.17 Non-CPU time-related metrics

- MIPS (millions of instructions per second)
- Consumed MIPS
- MFLOPS (millions of floating point instructions per second)
- CPU Service Units
- CP Service Rate
- Millions of Service Units per Hour (MSU/h)
5-minute 4-hour rolling average
- Internal Throughput Rate (ITR)
- Relative Processor Power (RPP)

Figure 2-17 Non-CPU time-related metrics

Non-CPU time-related metrics

Because CPU time is not repetitive across machines with different CPU speeds, several metrics have been introduced to solve this problem, such as consumed MIPS, CPU service units, and MSU.

Other metrics have been developed to measure CPU speed, as such MIPS, CPU Service rate, MSU/h, ITR, RPP, and others. However, all of them in the same machine vary with the set of instructions being executed.

2.18 Millions of instructions per second (MIPS)

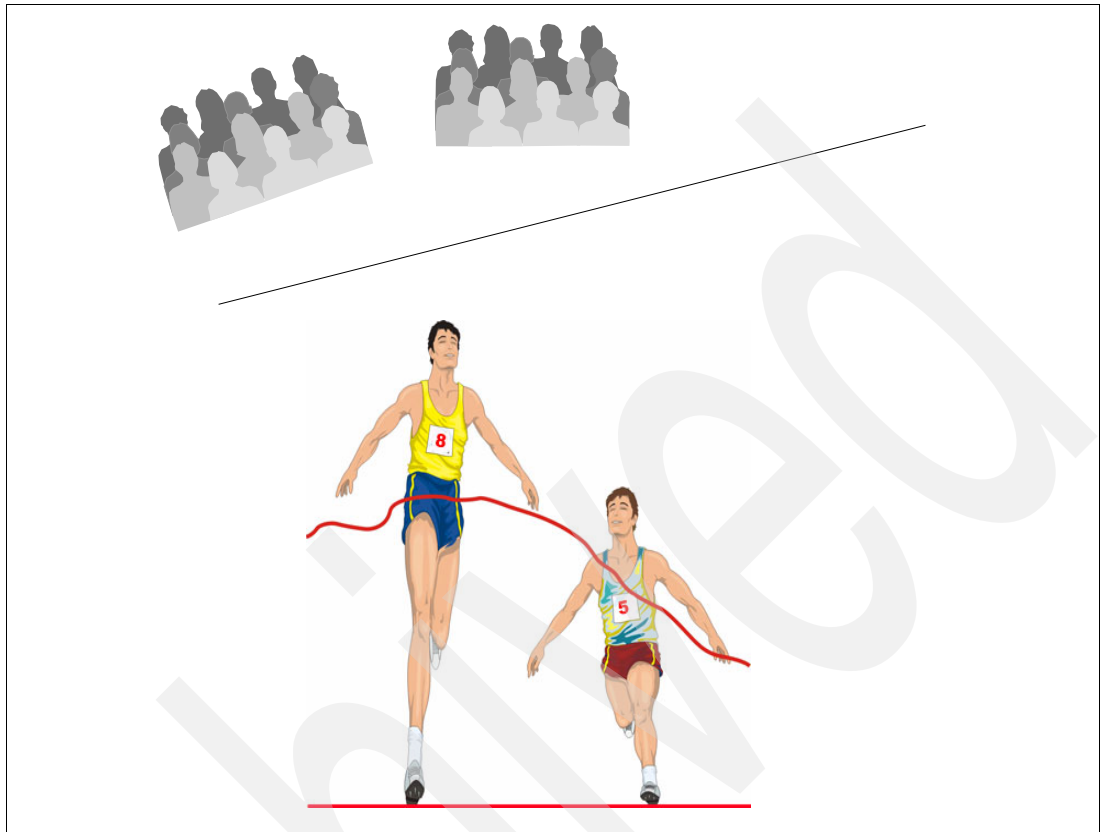


Figure 2-18 A way to think about comparing processor speed

MIPS as millions of instructions per second

The acronym MIPS has two perceived meanings. The first meaning has historical roots related to number of instructions executed per second, and it is associated with CPU speed. The second one (also called MIPS consumed) is simply a metric for CPU capacity that should not depend on CPU speed.

Note: Hardware performance ratings such as MIPS or MSUs indicate machine-wide throughput, and not the speed of a single processor. When analyzing how well an application has performed, there are potentially a large number of measurements that can be used to gauge workload efficiency in terms of the specific resources the application used. The most common resource used in measuring performance is time. Memory, disk space, and other resource usage are often used, too, because of their impact on performance.

Thus, comparing MIPS or MSU ratings between machines to estimate transformation speed will not produce reliable estimates if the number of processors differs. To derive MSU ratings that can provide valid estimates, the number of processors must be taken into account.

MIPS means millions of instructions per second. (MIPS is not the plural of MIP; for example, we say a processor has the speed of 1 MIPS, not 1 MIP.)

However, using MIPS as a comparison between two processors is fair if the processors execute exactly the same set of instructions when executing the logic associated with the transaction (dispatchable unit). When the instruction sets are the same, then the path lengths are the same and consequently, the processor using less CPU time has more MIPS.

When the instruction sets are *not* the same, one processor can execute more simple instructions and the other processor can execute more complex instructions. In this case, comparing two processors with the MIPS metric is like declaring the winner of a race to be the athlete who took more steps per second in a race, like the smaller runner in Figure 2-18. Because of that, some IT professionals think that MIPS translates into misleading information about processor speed.

Furthermore, the same processor can deliver different numerical values for MIPS for different programs depending on the set of executed instructions (from a few to a large number of MIPS).

Historically, using instrumentation IBM measured the MIPS of different programs running in distinct machines. Currently it is considered to be an imagined figure often based on vendor or consultant claims.

With the introduction of the CPU-measurement Facility (CPMF), hardware measurement instrumentation allows us to again measure program MIPS.

One variation of MIPS is MFLOPS (millions of binary floating point instructions per second). It is used in numerical-intensive computing (scientific and technical) only.

Consumed MIPS is just another metric associated with CPU time

Today the value of consumed MIPS is used as a metric to reflect relative processor capacity by a program and it is not associated with processor speed. To convert CPU time consumed by a transaction into consumed MIPS refer to 2.18, “Millions of instructions per second (MIPS)” on page 83. CPU consumed by a program depends on the CPU time for its execution and the currently defined machine MIPS (as discussed, an imagined value).

CPU time and MIPS consumed have the following differences when used to compare processors:

- ▶ CPU time is measured and consumed MIPS is calculated.
- ▶ CPU time varies when the transaction switches between processors of various speeds. MIPS *consumed* by one transaction theoretically do not vary with the speed of a processor.

2.19 CPU service units

- ❑ Formulas for CPU service units derivation:
 - Raw CPU Service Unit = CPU_TIME * SRM_Constant
 - CPU Service Unit = SDC * raw CPU Service Units
- ❑ CPU_TIME includes only productive time, and it has two parts:
 - TCB time in seconds (includes TCB time plus SRB enclave plus SRB client)
 - SRB time in seconds (includes non-preemptible SRB time only)

Figure 2-19 Formula for CPU service units derivation

CPU service units

Service units (SUs) are used by the System Resource Manager (SRM) as a basis for resource management. SRM calculates CPU and SRB service based on the task and SRB execution time, the CPU model, and the number of processors that are online. Tables describing the service consumed per second of execution time by CPU model are published in *z/OS MVS Initialization and Tuning Guide*, SA22-7591. The values listed for each processor are called *SRM constants*.

The SRM constant is a number derived by IBM product engineering to normalize the speed of a CPU, so a given amount of work will report the same service unit consumption on different processors. This removes the need to redesign your performance parameters each time you upgrade your CPU. Service units can be accumulated for CPU/SRB time, and I/O and processor storage.

Installations, through the use of installation-defined user coefficients, can give additional weight to one type of service relative to another. This allows the installation to specify which type of resource consumption should be emphasized in the calculation of service rates.

CPU service unit metric

CPU service units is a metric used by z/OS to measure the CPU consumption by transactions executing under z/OS processes, such as tasks (TCBs) or service requests (SRBs).

CPU service units are a measure of the task control block (TCB) execution time multiplied by an SRM constant that is CPU model-dependent. To introduce repetitiveness, the formula of raw CPU service units normalizes the CPU time as follows:

$$\text{Raw CPU Service Units} = \text{CPU_TIME} * \text{SRM_Constant}$$

The CPU_TIME refers to the measured values of TCB time and SRB time in seconds.

SRM constant (SUs/sec)

The SRM constant, in SUs/sec, is a number derived by IBM processor engineering to normalize the speed of a CPU, so that a given amount of work will report approximately the same service unit consumption on different speed processors. The higher the speed of the processor, then the higher the numeric value of the SRM constant.

The SRM constant is measured by IBM on different machines and different types of transactions. For published values of the SRM constant, see 2.70, “z10 EC SRM Constants” on page 173.

The faster the CPU and the less CPU time, then the more the SRM constant increases (to compensate for the lesser CPU time). Here is a numerical example:

$$\begin{aligned} \text{CPU A - CPU Service Units} &= \text{CPU time}(2 \text{ secs}) * \text{SRM Constant (1)} = 2 \text{ CPU SUs} \\ \text{CPU B - CPU Service Units} &= \text{CPU time}(1 \text{ secs}) * \text{SRM Constant (2)} = 2 \text{ CPU SUs} \end{aligned}$$

Increasing the number of CPUs decreases the SRM constant, due to the Multi-Processing (MP) effect. In LPAR mode, the SRM constant is calculated in each logical partition based on the number of logical CPs of that LPAR, not based on the number of physical CPs of the CEC.

Service definition coefficient

Because not all kinds of services are equal in every installation, you can assign additional weight to one kind of service over another. This weight is called a service coefficient. To allow installations to weight the amount of CPU service units consumed, the service definition coefficient was introduced in the WLM policy. The unweighted service units per second of task or SRB execution time is a measure of each CPU model, but is independent of the values of the service coefficients.

The final calculation to derive CPU service units is as follows:

$$\text{CPU Service Units} = \text{Raw CPU Service Units} * \text{Service Coefficient}$$

Keep in mind that raw CPU service units is the previous multiplication of CPU time by the SRM constant. There is one service coefficient per CPU TCB time (task, client SRB and enclave SRB dispatchable units) and one service coefficient for CPU SRB time (traditional SRB dispatchable unit).

Currently it is good practice to have the value of these service coefficients be one (1.0), thereby not modifying the final value. (Note that TCB service units are incorrectly referred to in RMF reports as “CPU service units”. SRB service units are correctly referred to as *SRB service units*.)

Figure 2-15 shows an extract of the RMF Workload Activity report. The TCB time consumed by the transactions is 152.0 seconds, and the corresponding CPU service units is 1377 K. The SRB time consumed by the transaction is 7.3 seconds and the corresponding SRB service units is 66581.

Note: Many installations charge users’ departments on the basis of CPU service units.

CPU service rate

The metric associated with CPU service units that intends to also measure the CPU speed is the CPU Service Rate, which is obtained by dividing the CPU service units consumed by the elapsed time. The faster the processor, the higher the CPU Service Rate.

The Large System Performance Reference (LSPR) numbers are based on an approximation of your own workload mix. Refer to 2.23, “LSPR relative processor power (RPP)” on page 94 for more information about this topic.

Archived

2.20 RMF Partition Data report

PARTITION DATA REPORT															PAGE 3	
z/OS VIR11			SYSTEM ID S59			DATE 11/28/2009			INTERVAL 15.00.010							
MVS PARTITION NAME			RPT VERSION VIR11 RMF			TIME 17.00.00			CYCLE 1.000 SECONDS			GROUP NAME				
IMAGE CAPACITY			1127			NUMBER OF PHYSICAL PROCESSORS			26			LIMIT				
NUMBER OF CONFIGURED PARTITIONS			12			AAP			2			AVAILABLE				
WAIT COMPLETION			NO			ICF			0							
DISPATCH INTERVAL			DYNAMIC			IIP			2							
----- PARTITION DATA -----																
NAME	S	MGT	DEF	ACT	DEF	WLM%	NUM	TYPE	EFFECTIVE	TOTAL	AVERAGE PROCESSOR UTILIZATION PERCENTAGES					
											LOGICAL PROCESSORS	PHYSICAL PROCESSORS				
											EFFECTIVE	TOTAL	LPAR	MGMT	EFFECTIVE	TOTAL
S59	A	801	0	502	NO	0.0	20.0	CP	02.13.34.022	02.13.34.604	44.52	44.53	0.00	44.52	44.53	
S50	A	500	0	0	NO	0.0	20.0	CP	00.00.00.000	00.00.00.000	0.00	0.00	0.00	0.00	0.00	
S51	A	100	0	53	NO	0.0	3.0	CP	00.13.58.918	00.14.00.016	31.07	31.11	0.01	4.66	4.67	
S55	A	101	0	68	NO	0.0	20.0	CP	00.18.01.114	00.18.01.538	6.01	6.01	0.00	6.01	6.01	
S58	A	999	0	493	NO	0.0	20.0	CP	02.11.06.315	02.11.06.763	43.70	43.70	0.00	43.70	43.70	
PHYSICAL																
TOTAL																
S59	A	150					2	AAP	00.00.00.373	00.00.00.419	0.02	0.02	0.00	0.02	0.02	
S50	A	150					2	AAP	00.00.00.000	00.00.00.000	0.00	0.00	0.00	0.00	0.00	
S51	A	150					2	AAP	00.00.00.737	00.00.00.770	0.04	0.04	0.00	0.04	0.04	
S55	A	150					2	AAP	00.00.00.283	00.00.00.327	0.02	0.02	0.00	0.02	0.02	
S58	A	150					2	AAP	00.00.00.317	00.00.00.359	0.02	0.02	0.00	0.02	0.02	
PHYSICAL																
TOTAL																
XSCF90A	A	DED					2	ICF	00.00.01.713	00.00.02.870	99.99	99.99	0.00	99.99	99.99	
PHYSICAL																
TOTAL																
S59	A	150					2	IIP	00.00.00.282	00.00.00.326	0.02	0.02	0.00	0.02	0.02	
S50	A	150					2	IIP	00.00.00.000	00.00.00.000	0.00	0.00	0.00	0.00	0.00	
S51	A	150					2	IIP	00.00.00.942	00.00.00.973	0.05	0.05	0.00	0.05	0.05	
S55	A	150					2	IIP	00.00.00.311	00.00.00.359	0.02	0.02	0.00	0.02	0.02	
S58	A	150					2	IIP	00.00.00.269	00.00.00.307	0.01	0.02	0.00	0.01	0.02	
PHYSICAL																
TOTAL																
TOTAL																

Figure 2-20 RMF Partition Data report

RMF Partition Data report

The Partition Data Report section of the CPU Activity report provides data about all configured partitions, independent of the type of operating system running in each partition. This section only appears when RMF is running in a Processor Resource/Systems Manager™ environment in LPAR mode, and is only for partitions active at the end of the duration interval. The section is not available if RMF is running in a z/VM guest environment.

RMF provides information about the partitions that are sharing the processor. If you want further information about a particular partition, you can run RMF separately on that partition.

This metric is the numerical division of CPU service units by 1-Mega. It produces smaller figures. MSUs are like CPU service units and consumed MIPS, and theoretically do not vary with the CPU speed.

Millions of CPU service units per hour (MSU/h)

This metric has the same meaning of CPU service rate (that is, per second) but with various scale units. It should vary with the CPU speed.

There are two types of MSU/h scales:

- ▶ Average MSU/hour used in PSLC
- ▶ 4-hour rolling average MSU used in WLC and WLM/SRM capping decisions

4-hour rolling average

A 4-hour rolling average MSU means that each instant value is the average of the last 4 hours. This value is recalculated every 5 minutes by adding the most recent 5-minute interval accumulated CPU service unit and discarding the oldest 5-minute interval.

The 4-hour rolling average allows you to have higher utilization within the 4-hour interval, which can be compensated by a valley with a lack of activity.

Currently the derivation of the MSU/h for one CEC is done through the LSPR project (see 2.23, "LSPR relative processor power (RPP)" on page 94), and it is not connected with the SRM constant figure.

The formula to convert CPU service units consumed in MSUs /h is as follows:

$$\text{MSUs per hour} = (\text{CPU_Service_Units} * 3600) / 1000000$$

Note: It is possible to convert, in one CEC, MSUs to MIPS. One MSU per hour is approximately 6 MIPS.

In the RMF report shown in Figure 2-20, the logical partition NP1 consumed 10 MSU/h along the RMF interval. The partition is software-capped in 100 MSU/h. The value 62.2% in the column CAPPING WLM% indicates that LP NP1 was soft-capped along the RMF interval.

2.21 Internal throughput rate (ITR)

□ ITR formulas:

- $\text{ITR} = \text{Number of ended transactions} / \text{CPU time consumed}$
- $\text{ETR} = \text{ITR} * \text{CPU utilization}$

□ ITR is a function of:

- CPU speed
- Operating system
- Transaction load characteristics

Figure 2-21 Internal throughput rate

Internal throughput rate (ITR)

Internal throughput rate (ITR) is another metric used to measure CPU speed. It determines the capacity of a processor in terms of the number of transactions per CPU second (CPU Time):

$$\text{ITR} = \text{Number of ended transactions} / \text{CPU time consumed}$$

ITR is a function of the following elements:

- ▶ CP speed (the faster the CP, the higher the ITR)
- ▶ The operating system (a well-designed operating system produces a higher ITR)
- ▶ Transaction workload characteristics (short, trivial transactions result in a higher ITR)

If the operating system (z/OS, for example) and transaction workload mix are constant, ITR can be used to measure CPU speed in terms of transactions. Comparatively, ITR is more suitable for such a comparison than ETR because ETR depends on many more variables. Refer to 2.9, “External throughput rate (ETR)” on page 69 for more detailed information about this topic.

Large Systems Performance Reference (LSPR)

IBM calculates ITRs for several workloads and machines by using the Large Systems Performance Reference (LSPR) methodology; refer to 2.23, “LSPR relative processor power (RPP)” on page 94 for details. ITR provides a reliable basis for measuring capacity and performance.

The relation between ETR and ITR is expressed as follows:

$$\text{ETR} = \text{ITR} * \text{CPU utilization}$$

When the CP utilization is 100%, the ETR is equal to the ITR.

LSPR considerations

The major problem of measuring CPU speed is that for any CEC, such speed depends much on the set of executed instructions. IBM Large Systems Performance Reference (LSPR) ratios represent an IBM assessment of relative processor capacity (through ITRs) in an unconstrained (no bottlenecks) environment for the specific benchmark workloads and system control programs specified in the tables. Ratios are based on measurements and analysis. The amount of analysis as compared to measurement varies with each processor.

Many factors, including but not limited to the following, can result in the variances between the ratios provided by LSPR and actual operating environments:

- ▶ Differences between the specified workload characteristics and your operating environment
- ▶ Differences between the specified system control program and your actual system control program
- ▶ I/O constraints in your environment
- ▶ Incorrect assumptions in the analysis
- ▶ Unknown hardware defects in processors used for measurement
- ▶ Different number of z10 books for the same amount of active PUs.

IBM does not guarantee that your results will correspond to the ratios provided at the LSPR. That information is provided “as is”, without warranty, express or implied.

LSPR data for IBM processors is generally made available at announce time. Refer to 2.23, “LSPR relative processor power (RPP)” on page 94 for more information about this topic. IBM announcement claims are based on LSPR measurements calculated by averaging the ITR in several workloads.

2.22 Large Systems Performance Reference (LSPR)

- ❑ Large Systems Performance Reference (LSPR) is an IBM project using the benchmark methodology to estimate the PU capacity needed for your workload.
- ❑ LSPR measures the internal throughput rate (ITR) of different machines with different workload types and different operating systems (z/OS, z/VM, VSE, z/ACP).
- ❑ It is based on the idea of comparing the same set of workloads on several CECs.

Figure 2-22 Large Systems Performance Reference

Large Systems Performance Reference (LSPR)

As previously mentioned, IBM Large System Performance Reference (LSPR) ratios represent an IBM assessment of relative processor capacity in an unconstrained environment for the specific benchmark workloads and system control programs specified in the tables. Ratios are based on measurements and analysis. The amount of analysis as compared to measurement varies with each processor.

Many factors, including but not limited to the following, can result in the variances between the ratios provided and actual operating environments:

- ▶ Differences between the specified workload characteristics and your operating environment
- ▶ Differences between the specified system control program and your actual system control program
- ▶ I/O constraints in your environment
- ▶ Incorrect assumptions in the analysis
- ▶ Unknown hardware defects in processors used for measurement

IBM does not guarantee that your results will correspond to the ratios provided herein. This information is provided “as is”, without warranty, express or implied. LSPR data for IBM processors is generally made available at announce time. IBM announcement claims are based on LSPR measurements.

LSPR benchmarks

IBM uses the LSPR benchmark for zSeries, z9, and z10 servers. When upgrading or buying a new machine, capacity planners are to always consider using LSPR tables to determine the possible model choice based on LSPR MIPS, or LSPR MSUs/hour or RPPs. The tools zPCR and zCP3000 are based on the same LSPR benchmark MIPS and the Internal Throughput Rate Ratios (ITRR) from the tables.

LSPR projections

Primary models of each processor series are always measured for the LSPR. It is impractical, however, for IBM to allocate the necessary resource to measure every individual processor model announced. For those that are not measured, ITR numbers are projected. A projected ITR is based on the known processor internal design, and on the known characteristics of the subject workload on similarly designed processor models. Projections have been shown to have accuracy comparable to that of measurements, where subsequent testing has occurred.

Note: Many older processor models (both IBM and IBM-compatible) that were measured in the past are no longer accessible for testing. It is highly desirable to maintain these processors in LSPR tables, for SCPs that are supported. Therefore, as the LSPR moves ahead to more current software levels, ITRs for these processors must be estimated. Estimates are based on known deltas between the older software and the current software on similarly designed processor models.

2.23 LSPR relative processor power (RPP)

zSeries 990 (zSeries 990 2084-301 = 1.0)

Processor	#CP	SMG	MSU*	Mixed	LoIO-Mix	TI-Mix	CB-L	CB-S	WASDB	OLTP-W	OLTP-T
2084-301	1	80	70	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
2084-302	2	IMLC	132	1.90	1.93	1.92	1.95	1.80	1.90	1.93	1.92
2084-303	3	IMLC	191	2.76	2.85	2.80	2.88	2.55	2.79	2.82	2.80
2084-304	4	IMLC	248	3.60	3.74	3.66	3.79	3.28	3.66	3.68	3.64
2084-305	5	IMLC	302	4.41	4.61	4.48	4.68	3.96	4.51	4.51	4.44
Ratios for 17 way and above are based on running multiple OS images.											
2084-317	17	IMLC	799	11.80	13.49	11.97	13.96	9.37	13.52	12.23	11.18
2084-318	18	IMLC	837	12.36	14.20	12.55	14.69	9.71	14.26	12.87	11.70
2084-319	19	IMLC	878	12.92	14.91	13.13	15.42	10.05	15.00	13.50	12.22
2084-320	20	IMLC	919	13.48	15.62	13.70	16.15	10.40	15.74	14.13	12.74
2084-321	21	IMLC	959	14.04	16.33	14.28	16.88	10.74	16.48	14.77	13.26
2084-322	22	IMLC	999	14.59	17.04	14.86	17.61	11.08	17.22	15.40	13.78
2084-323	23	IMLC	1037	15.15	17.75	15.43	18.34	11.43	17.96	16.03	14.30
2084-324	24	IMLC	1076	15.71	18.46	16.01	19.07	11.77	18.70	16.66	14.83
2084-325	25	IMLC	1114	16.26	19.17	16.58	19.80	12.11	19.44	17.30	15.35
2084-326	26	IMLC	1151	16.81	19.88	17.16	20.53	12.45	20.17	17.93	15.87
2084-327	27	IMLC	1188	17.37	20.59	17.73	21.26	12.80	20.91	18.56	16.39
2084-328	28	IMLC	1225	17.92	21.30	18.31	21.99	13.14	21.65	19.20	16.91
2084-329	29	IMLC	1261	18.48	22.02	18.88	22.72	13.48	22.39	19.83	17.43
2084-330	30	IMLC	1296	19.03	22.73	19.46	23.45	13.83	23.13	20.46	17.95
2084-331	31	IMLC	1332	19.58	23.44	20.03	24.18	14.17	23.87	21.10	18.48
2084-332	32	IMLC	1365	20.13	24.15	20.61	24.91	14.51	24.61	21.73	19.00

Figure 2-23 LSPR table figures for z990

LSPR relative processor power (RPP)

To determine the transaction CPU time, you need to have a real run of such a transaction. This is called benchmarking. To address that problem, IBM implemented a benchmark method to evaluate CPU capacity. The project is called Large Systems Performance Reference (LSPR). It measures the ITRs of different machines with different workload types and different operating systems (z/OS, z/VM, VSE, ACP/TPF). After that, the ITRs derived are normalized (a ratio is calculated) to a certain base machine (2084 - 301 for example), as shown in Figure 2-23 on page 94. The result of this normalization is called RPP (or ITRR, the Internal Throughput Ratio).

The column MSU refers to MSU/h.

LSPR ratios

LSPR ratios represent the IBM assessment of relative processor capacity in an unconstrained environment for the specific benchmark workloads and system control programs specified in the tables. Ratios are based on measurements and analysis. The data shown in Figure 2-23 on page 94 is based solely on IBM measurements and analysis of the processors in the tables.

LSPR workloads

Each individual LSPR workload is designed to focus on a major type of activity, such as interactive, online database, or batch. The LSPR does not focus on individual pieces of work such as a specific job or application. Instead, each LSPR workload includes a broad mix of

activity related to that workload type. Focusing on a broad mix can help assure that resulting capacity comparisons are not skewed.

The workloads include:

- ▶ OLTP-T (formerly IMS) - Traditional On-line Workload
- ▶ OLTP-W -Web-enabled On-line Workload
- ▶ WASDB - WebSphere® Application Server and Data Base
- ▶ CB-L (formerly CBW2)-Commercial Batch Long Job Steps
- ▶ CB-I (formerly CB84)-Commercial Batch Short Job Steps
- ▶ CICS/DB2 - On-line Workload for pre-z/OS Version 1 Release 4
- ▶ CICS - On-line Workload prior to OS/390 Version 2 Release 10 only
- ▶ DB2 - On-line Workload for OS/390 Version 1 Release 1 only
- ▶ TSO - Online Workload for pre-z/OS Version 1 Release 4
- ▶ FPC1 - Engineering/Scientific Batch Workload for pre-z/OS Version 1 Release 4

Relative processor power (RPP)

Relative processor power (RPP) is the average ITR ratio of the workloads (the Mixed column shown in Figure 2-23). This column is used to evaluate for the specific machine the metrics associated with CPU speed, such as: SRM constant, MIPS, MSUs/h, and the M value in the RIOC formula.

RPP is the ratio of ITRs for a specific workload mix (25% of each) for different machines. RPP is normalized to a base machine. So, if a given transaction takes a certain amount of RPPs, you can estimate how much it will consume on a different machine by using the following formula:

$$\text{Utilization (New CPU)} = \text{RPP(CPU old)} / \text{RPP(CPU new)} * \text{Utilization(CPU old)}$$

In summary, various numbers are commonly be used as approximate indicators of CPU capacity. Remember that most of these are general approximations only, and your actual capacity can vary significantly. To obtain the most accurate assessment of CPU capacity for your workload, you must take into account differences in workload processing characteristics by using a methodology like that described in the Large Systems Performance Reference. You can find the most current version at the following site:

<http://www.ibm.com/servers/eserver/zseries/lspr/zSeries.html>

It is best practice for each client to build its own LSPR ratios based on the workload mix that the installation is running. This methodology is described at the following sites:

<http://www-1.ibm.com/servers/eserver/zseries/lspr/lsprmixwork.html>

<http://www-03.ibm.com/support/techdocs/atmastr.nsf/WebIndex/PRS135>

2.24 CP-measurement facility highlights

- ❑ Hardware technology enhancements have not kept pace with business required system performance improvements and we are reaching physical limits
- ❑ A key area for performance improvement is more synergy between hardware and software
- ❑ CPU-measurement facility (CPMF) is a z10 hardware instrumentation function
 - CPMF gathers performance data on hardware (counters and sampling), which supports software performance optimizations
- ❑ Hardware measurements are superior to software measurements

Figure 2-24 CPMF highlights

CPU-measurement facility (CPMF) highlights

Recently, hardware technology enhancements have not kept pace with business requirements for system performance improvements. Solutions that involve more than just faster hardware are needed to fill the gap. A key area for performance burst is more synergy between hardware and software (operating system, middleware, compilers, and client applications).

To address this issue, IBM has introduced System z10 hardware instrumentation, the CPU-processor measurement facility (CPMF), which supports software performance optimizations by providing counters and sampling about hardware functioning. It is a nondisruptive data gatherer with low overhead, and it can run in multiple logical partitions and virtual machines simultaneously.

CPMF is built into the PU and allows software to be asynchronously informed about PU activities to determine hot spots. It helps to understand why a particular application is slow and whether optimizations achieve the desired effect. This makes understanding performance problems easier and much more precise than the software-based performance data gatherer methods used previously. If the parametrization is correct (for example, avoid CPMF sampling mode), then the new hardware instrumentation does not disturb the system being observed (with nearly no impact), although it requires some support from the operating system. For example, z/OS allocates real storage in which the PU running CPMF can store measurement data.

CPMF measurement and performance data

Measurement data provided by the CPMF is intended for statistical estimation of performance characteristics. This instrumentation is virtualized so that it can run in multiple logical partitions concurrently producing data at the logical PU level. When a CPU logical is in wait state, the instrumentation stops.

CPMF performance data may also help clients in redesigning performance with the hardware by assisting in the following decisions:

- ▶ Deciding on how many books in the CEC (regarding performance)
- ▶ Deciding on how many PUs per type (CP, IFL, zIIP) per book
- ▶ Determining how the mixing of different PUs types can affect performance
- ▶ Determining how to distribute main storage across the books
- ▶ Determining the performance limit of logical PUs in a z/OS running the real workload
- ▶ Determining the better mixing of workloads per z/OS to obtain better performance
- ▶ Evaluating the CPU performance effect of real multiprocessing in a CICS address space through the use of CICS Safe Threads functionality
- ▶ Determining whether to implement DB2 1-mega pages to improve the use of TLBs in virtual storage DAT performance (see “Translation lookaside buffer (TLB)” on page 127 for more information about TLBs)

Using CPMF instrumentation

This section provides an example demonstrating how you can use CPMF instrumentation data in a real-life situation.

- ▶ A client reports a performance problem on a z10 system.
- ▶ The problem is routed to a System Center or other performance team.
- ▶ An IBM performance expert asks the client to collect instrumentation data to help resolve the issue:
 - If diagnosis sampling is needed, an IBM representative with a PE password can enable this (see “CPU-measurement sampling facility functions” on page 150 for more information about diagnosis sampling).
- ▶ The IBM performance expert post-processes and analyzes the results.
- ▶ The IBM performance expert provides feedback to the client.

2.25 Hardware and software requirements for CPMF

- ❑ System z10 machine must be at minimum LIC level
 - Check with IBM for current level
 - z10 BC sub-capacity models supported
- ❑ z/OS LPAR being measured must be at z/OS V1R8 or higher with APARs:
 - OA25755, OA25750, and OA25773
- ❑ Not supported for z/OS running as a z/VM guest
- ❑ The CPU Measurement Facility is designed to:
 - Increase the ability to run hardware instrumentation
 - Support application performance tuning

Figure 2-25 Hardware and software requirements for CPMF

Hardware and software requirements for CPMF

Figure 2-25 on page 98 shows the hardware and software requirements to run CPMF instrumentation in one z/OS system. This function provides a framework for collecting and recording hardware event data for IBM System z10 machines with MCL stream SYSTEM, EC number N10970 firmware. APARs OA25750, OA257rr, and OA25773 are required to enable full function of this support.

Hardware and software requirements

The most important requirements to remember are:

- ▶ CEC: Z10 EC and z10 BC
- ▶ Minimum z/OS release: z/OS V1R8 with PTFs

Hardware-based performance analysis (CPMF)

Hardware instrumentation offers many benefits over the software-based performance tools used in previous systems. The CPU Measurement Facility (CPMF), available in z10 systems, provides support built into the processor for performance analysis. Exploiting this mechanism allows you to observe the performance behavior of an operating system running client applications, and it does so with little impact to the system being observed.

The CPU Measurement Facility is designed to increase the ability to run hardware instrumentation in the field and to support application performance tuning.

System z hardware instrumentation

In the past, System z mainframe hardware instrumentation has been used for the following tasks:

- ▶ Debugging hardware performance and enabling performance tuning by the sophisticated compilers of operating systems and internal subsystems
- ▶ Allocating (or deallocating) a sufficient hardware system area for storing measurement data

However, allocation or deallocation of hardware system area requires a machine power-on. Also, hardware instrumentation runs on CPs at the basic machine-level only; thus, running it requires the entire machine. Because of these constraints, hardware instrumentation has been limited mainly to laboratory use. The CPU Measurement Facility resolves these issues by enabling the control program to allocate real storage in which the machine can store measurement data, and to virtualize the hardware instrumentation so that it can run in multiple LPARs concurrently.

The format and meaning of measurement data used to facilitate application performance tuning are defined in the architecture. The remainder of the CPU Measurement Facility is used to establish controls and extract measurement data.

2.26 CPMF counters and sampling

❑ Counters used to count real time PU activities include:

- Instruction count
- Cycle count
- Cache events
- TLB events

Counters are grouped into counter sets

❑ Sampling data - Each sample data includes:

- Instruction address
- Primary address space number
- State information about the PU

Figure 2-26 CPMF counters and sampling

CPMF counters and sampling

The CPMF is available in each logical PU. It gathers data that can be used to understand performance problems and to analyze the effect of optimizations. The CPMF captured data PU consists of two parts:

- ▶ Counters are used to count real-time PU activities. Data is provided by the CPMF component known as the CPU-measurement counter facility. There is no measurable overhead in capturing such information. It is a best practice to gather such counts long term active. The captured data includes:

Instruction count	The number of executed instructions
Cycle count	The number of cycles needed to execute them
Cache events	The number of Directory Writes and Penalties
TLB events	The number of translation entries written and misses to TLBs

For more information about the CPU-measurement counter facility, see 2.28, “CPU-measurement counter facility” on page 103.

- ▶ Sampling data is provided by the channel path measurement facility component. The set of performance data produced by this facility covers detailed and frequent events. Instead of having a real counter, it uses a sampling technique.

It provides a means to take a snapshot of the PU at a program-specified time interval. At the end of each interval, the PU stores sample data into measurement blocks allocated by the operating system. A frequent sampling (small time interval) may cause some CPMF

overhead, but much less than a sampling software performance monitor. Each sample data includes the following:

- Instruction address
- Primary address space number (ASID number)
- State information about the CP, such as privileged or problem mode, disabled for interrupts or enabled, and PSW key value

This allows tooling programs to map instruction addresses into load modules or tasks, and it facilitates tracing of hot spots. It is the same concept as the well-known Program Event Recording (PER) with the Instruction-fetch event on. The major difference is that PER generates a program interrupt per every executed instruction, thereby incurring heavy overhead.

For more information about the CPU-measurement sampling facility, see 2.55, “CPU-measurement sampling facility” on page 149.

For both counters and sampling, provisions have been made to separate the measured data between problem-state and privileged-state PU activities so that application software and system software can be independently optimized.

When either the CPU-measurement counter or sampling facility is installed, it includes an external-interruption subclass known as the measurement-alert subclass. For measurement-alert interruptions, a 32-bit parameter is associated with the interruption and is stored at real locations 128-131 at PSA.

2.27 Set of instructions to interface with CPMF

- ❑ The following instructions activate and manage the CPU-measurement counter facility:
 - EXTRACT COPROCESSOR-GROUP ADDRESS
 - EXTRACT CPU COUNTER
 - EXTRACT PERIPHERAL COUNTER
 - QUERY COUNTER INFORMATION
 - SET CPU COUNTER
 - SET CPU-COUNTER-SET CONTROLS
 - SET PERIPHERAL COUNTER
 - SET PERIPHERAL-COUNTERSET CONTROLS
- ❑ The following instructions activate and manage the CPU-measurement sampling facility:
 - SET PROGRAM PARAMETER
 - QUERY SAMPLING INFORMATION
 - SET SAMPLING CONTROLS

Figure 2-27 Set of instructions to interface with CPMF

Set of instructions to interface with CPMF

Figure 2-27 on page 102 shows all the z/Architecture instructions used by z/OS HIS to interface with CPMF.

Those instructions are privileged but some of them may be accessed in problem state by, for example, performance monitors depending on a bit set in control register 0.

We describe these instructions here simply to clarify the following hierarchy:

- ▶ System programmers manage z/OS HIS.
- ▶ z/OS HIS manages CPMF through the referred instructions.
- ▶ CPMF produces data according to the instructions.
- ▶ z/OS HIS consolidates the data in UNIX System Services files and SMF 113 records.
- ▶ This data is then processed by performance monitors.

2.28 CPU-measurement counter facility

- ❑ A basic counter set contains:
 - Number of executed cycles, number of executed instructions and L1 cache basic sourcing information
- ❑ A problem state counter set contains:
 - Number of cycles, number of instructions, L1 cache basic sourcing information is only in problem state
- ❑ A crypto activity counter set contains:
 - Counts and cycles by crypto CPACF function
 - This counter set is a coprocessor-group counter set
- ❑ Extended counter set counters:
 - Future compatibility with new machines is not honored
 - In a z10 they contain more detailed z10 cache hierarchy and virtual addressing TLB information

Figure 2-28 CPU-measurement counter facility counter sets

CPU-measurement counter facility

The CPMF enhancement is available in each CP. It gathers data that can be used to understand performance problems and to analyze the effect of optimizations. The CPMF in each CP consists of two parts: counters and sampling.

Counters are used to count CP activities, including instruction count, cycle count, and cache events. Counter values allow the program to quickly compute a performance estimation, such as cycles per instruction or average cache penalty, to determine whether performance problems exist. Sampling provides a means to take a snapshot of the CP at a program-specified sampling interval. At the end of each sampling interval, the machine stores sample data into measurement blocks allocated by the control program.

Each sample data includes an instruction address, the primary address space number, and some state information about the CP. This allows tooling programs to map instruction addresses into modules or tasks and facilitates the determination of hotspots. For both counters and sampling, provisions have been made to separate the measured data between problem-state and privileged-state CP activities so that application and system software can be independently optimized.

Counters are grouped into counter sets. Counter controls are defined at the counter set level. For example, counters are grouped into counter sets. Counter controls are defined at the counter set level; when the operating system activates a counter set, all counters in the set are activated. Each counter set can also be individually activated.

A counter set is provided to count PU activities in only the problem state, and another is provided for PU activities in both the problem and the privileged state.

The facility provides two kinds of counters:

- ▶ Local to the PU, and grouped into several PU counter sets
- ▶ Global to all PUs, and used to count activities of peripheral coprocessors, which are shared among certain CPUs and are grouped into different peripheral counter sets

The facility provides the following PU counter sets on each PU:

- ▶ A basic counter set, where IBM guarantees compatibility with future machines.

This counter set contains the number of executed cycles, the number of executed instructions, and L1 cache basic sourcing information. It also includes the processor speed for which the hardware event counters are recorded. This speed is measured in cycles/microsecond (a type of frequency). This function is activated by setting the authorization control in the HMC.

- ▶ A problem state counter set, which is a subtype of the basic counter.

This counter set contains the number of cycles, the number of instructions, and L1 cache basic sourcing information, but all this information only in problem state, that is, when bit 15 is on in the PSW. This function is activated by setting the authorization control in the HMC.

- ▶ The crypto activity counter set, which contains counts and cycles by crypto CPACF function.

See 2.49, “Hardware crypto in a z10” on page 140 to learn more about CPACF. This counter set is a coprocessor-group counter set. A coprocessor-group consists of a number of coprocessors that may be attached to one or more PUs.

An example of this in a z10 can be the CPACF crypto processor attached (shared) to two PUs. One coprocessor-group counter set is provided for each coprocessor-group. Counters in a coprocessor-group counter set measure the activities of each coprocessor in the group at the basic machine level. These functions (crypto and coprocessor-group) are activated by setting the authorization control in the HMC.

- ▶ The extended counter set, in which the future compatibility of these counters with new machines will not be honored; they are model-dependent.

Software processing CPMF extended counter data cannot expect these counters to remain unchanged when moving from one machine family to another. In a z10, extended counters contain more detailed z10 cache hierarchy and virtual addressing TLB information. This function is activated by setting the authorization control in the HMC.

PU counter set states

Each PU counter set can be in one of the following states:

- | | |
|---------------------|--|
| Unauthorized | When a counter set is in the unauthorized state, counters in the set cannot be used and do not increment. |
| Disabled | When a counter set is in the disabled state, the counter set is authorized for use but the program has not enabled the counter set yet. In this state, counters in the counter set do not increment and the counter contents cannot be extracted or set. The contents of all counters in a set are cleared to zeros (0) when the set is disabled. All PU counter sets are disabled by initial CPU reset, clear reset, or power-on reset. |

Inactive	In the inactive state, counters in the counter set do not increment; the counter contents remain unchanged and can be extracted or set.
Active	When a counter set is in the active state, the counter set is authorized, enabled, and activated. In this state, each counter in the set increments for the defined activity; the counter contents can be extracted or set.

Each CPU counter set can be enabled or disabled, and activated or deactivated by executing SET CPU-COUNTER-SET CONTROLS. Counters in the PU counter sets count PU activities on the logical PU basis.

2.29 How CPMF works

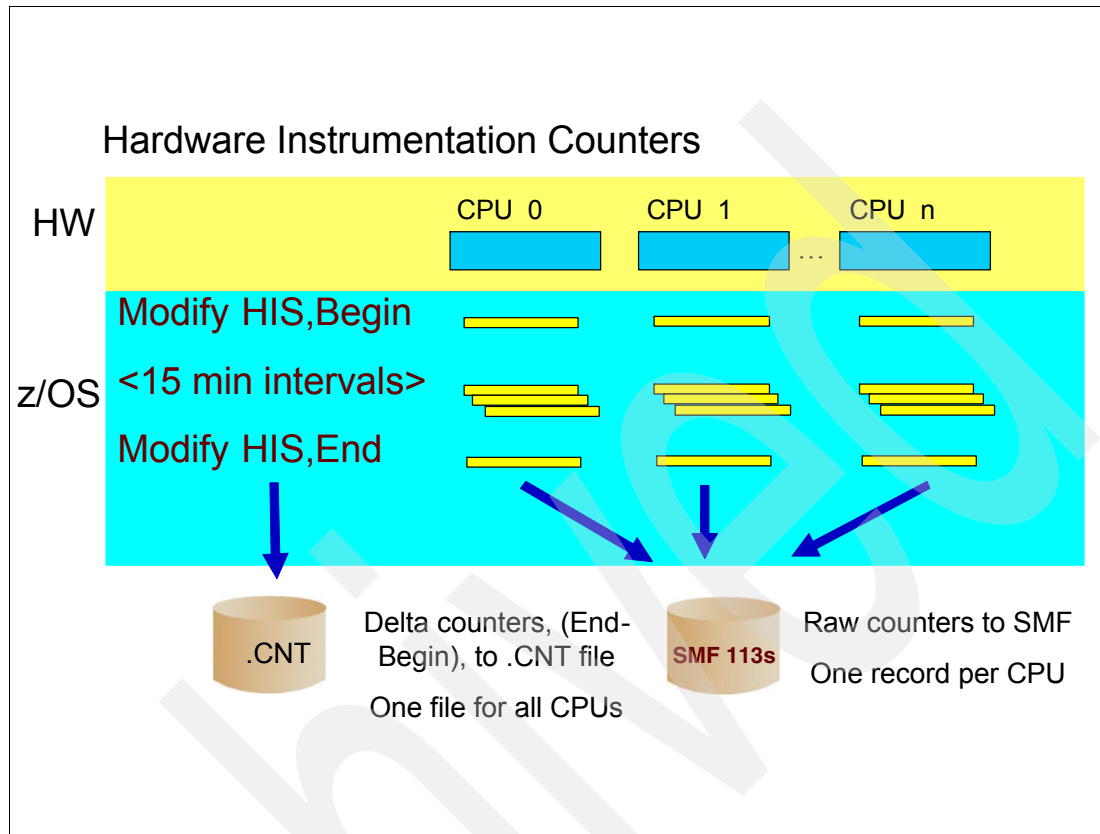


Figure 2-29 How CPMF works

How CPMF works

In z/OS there is a new component called Hardware Instrumentation Services (HIS) in charge of the communication with CPMF. HIS has its own PROCedure and occupies its own address space, by means of a **START** command.

Figure 2-26 shows the HIS being activated through a **MODIFY** console command. As a consequence HIS starts CPMF, which runs in several logical CPUs and captures counter and sample types of hardware performance data.

CPMF counter facility

The CPU measurement counter facility on each CPU consists of a local counter-set-state control register, a number of counters, several external-interruption events, a measurement-counter-extraction-authorization control, and a number of instructions, shown in 2.27, "Set of instructions to interface with CPMF" on page 102.

Note: CPMF functions (CPU-measurement counter facility and the CPU-measurement sampling facility) are to be disabled when they are not being used.

In the example pictured in Figure 2-26, CPMF is started and stopped through a **MODIFY HIS** command. HIS then generates a zFS file named **.CNT** with delta values for all CPUs. Also, after each 15-minute interval, HIS generates SMF type 113 records (one per each CPU).

2.30 Setting up hardware data collection

- ❑ HIS collects data for z10 processors
 - Writes data to z/OS UNIX file and SMF record 113
- ❑ Security requirements
 - Requires a user ID
 - An OMVS segment
 - A user HOME directory
- ❑ Enable SMF recording
 - Collect SMF record type 113 subtype 2
- ❑ START HISPROC
 - Stop data collection
 - F HISPROC,END

Figure 2-30 Configuring HIS

Hardware Instrumentation Services address space

The Hardware Instrumentation Services (HIS) address space must be started on each system where you want to collect data. Then you must configure and activate HIS data collection (hardware event counters and sampling facilities) for each system by issuing the **F HISPROC** command.

Important: Assign a sufficiently high dispatch priority to the instrumentation started task `hisproc`, so that the task can write sampling data to the SMP output files in a timely manner.

Setting up hardware event data collection

Hardware Instrumentation Services (HIS) is a function that collects hardware event data for processors at the System z10 Enterprise Class level or later.

During a run of hardware data collection, the system writes the data to a UNIX System Services output file and to SMF record type 113, subtype 2. The system writes the raw data to SMF record type 113 at 15-minute intervals during the data collection run, and writes the delta data to the UNIX System Services output file at the end of the run. The delta data is the data from between when the instrumentation run was initiated and the end of the run, showing the delta incremental values of the instrumentation run on the specified processor.

Security specifications

Before you start the HIS data collection, you may first need to authorize to the sampling facilities and counter set types you want to use through the support element (SE) console. For information about how to set up the authorization of the sampling facilities and counter sets, see *Support Element Operations Guide for System z10 machine* on the Resource Link™ home page at:

<http://www.ibm.com/servers/resourceLink>

Setting up hardware event data collection

Follow these steps to set up hardware event data collection:

1. Define a user ID for the HIS started task *hisproc*. Define the user ID with an OMVS segment that specifies:
 - Any UID.
 - A default HOME directory. For example, you might define the user ID as follows:
 - `ADDUSER hisproc OMVS(UID(25) HOME('/user'))`

2. Create a user HOME directory (in a local filesystem) by issuing the following `mkdir` command under OMVS:

```
mkdir /hisuser
```

Then assign read/write/exec authority to the `/hisuser` directory with the following `chmod` command:

```
chmod 007 /user
```

3. Enable SMF record type 113 using either the `SET SMF` or `SETSMF` commands. For example, you might enable SMF record type 113 as follows:

- Issue `SET SMF=xx` to select the SMFPRMxx parmlib member you want to update with information for SMF record type 113.
- Reply to the message issued in response to the `SET SMF=xx` command to change any SMFPRMxx parameters. For example, you might reply with the following information:

```
nn,sys(type(113)),intval(01),maxdorm(0100)
```

Reply `nn,U` to continue.

This reply will prompt the system to collect type 113 records, give you 1-minute collection intervals, and the data will only stay in the buffers for 1 minute before being written to the MANA or MANB data set. You can change other SMFPRMxx parameters on the `SET SMF` command response or the `SETSMF` command.

4. Start the HIS started task with the following command:

```
START HISPROC
```

This command does the following:

- ▶ It starts the Hardware Instrumentation Services (HIS) address space for the system.
- ▶ It creates a new instrumentation started task, *hisproc*, for the system.

5. Configure and start a run of data collection on a system with the following command:

```
F HISPROC,BEGIN
```

Note: You must explicitly start each run of hardware data collection. You cannot set up data collection to start running automatically.

2.31 PU caching z10 highlights

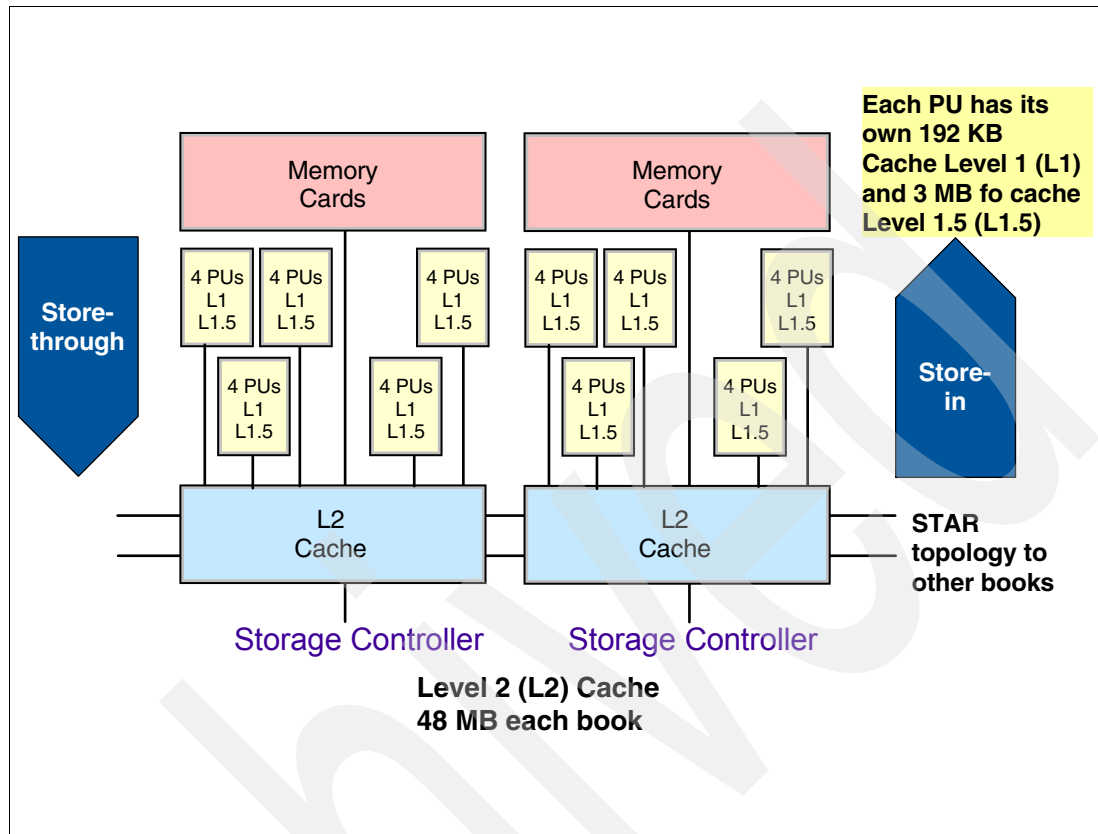


Figure 2-31 PU caching in z10

PU caching in z10 highlights

Before explaining CPMF caching counters events, we review here the concepts of PU caching in the z10. Caches are fast memories used to avoid the access to lower level memories that are slower, cheaper, and larger. In z10 EC there are three levels of cache (L1, L1.5, and L2) to improve performance by reducing access to real storage (called L3). Traditionally, PU caches are not directly visible to programs.

Each PU has its own 192 KB Cache Level 1 (L1), split into 128 KB for data (D-cache) and 64 KB for instructions (I-cache). The reason for splitting data and instructions is that the pattern and nature of the reference is different; for example, data is often changed, but instructions are seldom changed. The PU only fetches and stores instructions or operands (data) from or to the L1 cache. L1 cache is designed as a store-through cache, meaning that altered data is immediately (synchronously) stored to the next level of memory.

Cache levels

The next level of memory is the L1.5 cache that is in each PU and is 3 MB in size. It is also a store-through cache to L2.

L1 and L1.5 caches are PU private caches. L2 is a book shared cache (shared by all PUs of the same book).

L2 cache

Each book has a Level 2 (L2) cache of 24 MB for a total of 48 MB. Each L2 cache has a direct path to each of the other L2 caches in remote books on one side and each of the PUs in the book on the other side.

Then, L2 cache is shared by all PUs within a book. SC processor services provide the intelligence for the communication between L2 caches across books. L2 cache has a store-in buffer design, which means that the changes are not immediately copied to main storage (L3). The destage to main storage will be done when L2 occupancy reaches some threshold occupancy. The performance advantage of being store-in is that the PU does not need to wait for a slow store in memory. Also, if the same data is updated a few times, then when in an L2 cache, the store to main storage is performed only one time.

2.32 z10 cache performance aspects

- ❑ Cache performance depends on:
 - Cache technology.
 - Distance from the PU.
 - Size (larger the worse because of directory long searches).
 - Algorithms for keeping the best data (lines) in cache.
- ❑ Line (block) size at all cache levels on z CEC is 256 bytes.
- ❑ Caches are of two categories: private (L1 and L1.5) and shared (L2). Each level of the cache hierarchy is a superset of the subordinate caches; that is:
 - Every line in L1 cache also exists in associated L1.5 and L2.
 - Every line in L1.5 cache also exists in the associated L2.
- ❑ This simplifies coherency protocols and minimizes latencies. If the PUs share an L2 cache, only one copy of the line is needed in the L2.

Figure 2-32 z10 cache performance aspects

z10 cache performance aspects

As shown in 2.32, “z10 cache performance aspects” on page 111, cache performance depends on:

- ▶ Cache technology
- ▶ Distance from the PU
- ▶ Size (the larger, the worse because of directory long searches)
- ▶ Algorithms for keeping the best data (lines) in cache

Every new machine with a shorter cycle forces the cache to respond quicker to avoid creating bottlenecks. Access to large and distant caches cost more short cycles. This phenomenon of shrinking cache sizes can be seen in the design of the z10 EC, where the L1 instruction and data caches have been shortened (compared with z9) to accommodate the decreased cycle times.

The distance to remote caches also becomes a concern, for example, when an L2 cache is not located in the PU or, even worse, may be located in another book. To address that situation an L1.5 cache has been introduced in z10 EC, whereby the increase in latency (a big cache) is compensated for by the insertion of an intermediate level cache (less distance) reducing traffic to and from the L2 cache. A request is sent to L2 only if there is a cache miss in L1 and L1.5. See 2.33, “Cache page states” on page 113 for information about the relative distance of the cache and main storage in a z10 EC machine.

Line (block) size

When a line is first referenced by a CP, the line is installed in the associated shared L2 cache and in the L1.5 and L1 caches.

Each level of the cache hierarchy is a superset of the subordinate caches, that is:

- ▶ Every line in an L1 cache also exists in associated L1.5 and L2 caches.
- ▶ Every line in an L1.5 cache also exists in the associated L2 cache.

This simplifies coherency protocols and minimizes latencies. If the PUs share an L2 cache, only one copy of the line is needed in the L2.

Using cache in a z10

When the amount of available space is less than a threshold, the z10 EC uses a least recently used (LRU) algorithm to steal from the L1 cache the least referenced set of lines containing operands and instructions. The same LRU is also used from L1.5 and from L2 to main storage (L3). When a line “ages out” (LRU) of an L1 cache, it still exists in the L1.5 cache. When that line ages out of the L1.5 cache, it still exists in the L2 cache. For exclusive lines, the L1.5 cache sends a signal to the L2 cache that the line no longer exists in the L1.5 cache.

Every time that an operand or an instruction needs to be fetched, the L1 cache is inspected first. It is called a “hit” when the operand or instruction is found in the cache, and a “miss” when it is not found. In the rare case of a miss (less than 10%), we have the following sequence of searching: L1.5, L2 (same or different book) and main storage.

Cache differences

Besides the location, there are other differences between the L1/L1.5 cache and the L2 cache which make L1/L1.5 even faster than L2.

For the L1/L1.5 cache:

- ▶ There is error detection through a parity bit.
- ▶ There is no local error correction (data is recovered from L2).

For the L2 cache:

- ▶ There is error detection through ECC.
- ▶ There is local error correction (which makes it slower).

2.33 Cache page states

- ❑ A single cache line has one of two basic states:
 - Shared can exist simultaneously in the caches of more than one PU
 - It cannot be updated or modified (when shared). If the line resides in the private caches, there are several copies.
 - If the PUs share an L2 cache, only one copy is needed.
 - Exclusive to one PU and, therefore, that PU may modified. If an exclusive line is referenced by another PU, the state of the line is changed to shared.
 - If any stores (modifications) are pending, they must first complete before the line can be copied to the other CP.
- ❑ If a line exists in an L2 cache in the shared state and another PU requests the line, in a shared state as well, the L2 cache can honor the request immediately:
 - No pending stores are possible on this line.

Figure 2-33 Changing line caches in a z10

Changing the contents of L1 cache

When a PU executes an instruction that changes the contents of the output operand in its L1 cache, the following actions are executed:

- ▶ Post the other PUs, which may have in their L1 or L1.5 cache an out-of-date copy of the line changed by the first PU. In the z10 EC, the system controller (SC) located in each book has the L1 and L1.5 cache directories of all PUs of its book. When a PU alters the contents of one of its L1 cache lines, the SC is informed to invalidate this line in the L1 cache directory of other PUs in the same book. This update information is passed out to the other SCs (in other books) to execute the same invalidation, if needed.
- ▶ Define a place where the other PUs can fetch the most updated copy of these contents, if they need to. In a z10, the idea is to copy synchronously (store-through) the updated contents from L1 cache to L1.5 and from L1.5 to L2 a global memory where the other PUs can reach this most updated copy of the element.

The L2 cache marks the exclusive line “exclusive to no CP.” If another CP requests this line, the holding L2 cache can immediately honor the request for this line because no stores are pending in the L2 cache and no L1 cache or L1.5 cache is holding the line. Although most instructions are never modified and therefore exist as shared in all caches, most data is eventually updated, requiring the line to be held exclusive during the modification. A change in a shared page implies that all other copies are to be invalidated and the altered copy be declared as exclusive. A subsequent fetch of an exclusive line by another CP cannot be honored immediately by the L2 cache holding the line because coherency protocols must be followed.

2.34 Using CPMF caching counters data

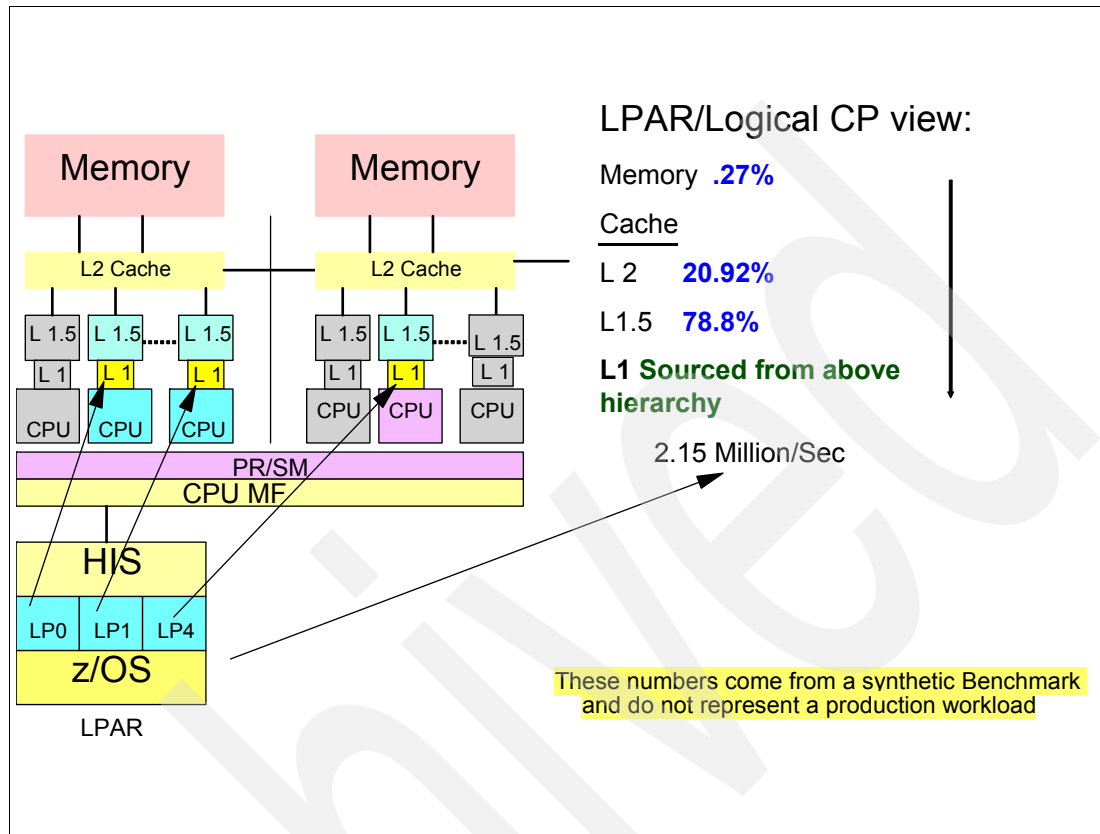


Figure 2-34 An example of caching counters

Using CPMF caching counters data

The CPU-measurement counter facility produces z10 cache information in two counter sets:

- ▶ A basic counter set with basic L1 cache information
- ▶ An extended counter set with more detailed hierarchy z10 cache information, including performance information about L1.5 and L2 caches

The example shown in Figure 2-34 on page 114 illustrates the following points:

- ▶ 78.80% of the misses in L1 are hits in L1.5.
- ▶ 20.92% of the misses in L1 are hits in L2 (local and remote).
- ▶ 0.27% of the misses in L1 are accessed in main storage (local and remote).

Cache basic counters

The basic counter set contains the following cache data:

- ▶ L1 Instruction - Cache Directory Write Count: the total number of L1 instruction cache directory writes. Directory Write means the cache directory is updated because either:
 - The cache line is brought in for a cache miss.
 - The state of a cache line is changed.

Note that some cache misses may not result in a directory write. For example, cache misses caused by incorrect branch prediction may not result in a directory write if the branch is resolved in time.

- ▶ L1 Instruction-Cache Penalty Cycle Count: the total number of cache penalty cycles spent loading the L1 instruction cache. The value of a cache-penalty cycle count divided by the value of the corresponding cache-directory-write count gives the average cache penalty per cache directory write.
- ▶ L1 Data-Cache Directory Write Count: the total number of L1 data-cache directory writes.
- ▶ L1 Data-Cache Penalty Cycle Count: the total number of cache penalty cycles spent for loading for L1 data cache.

Cache extended counters

For all counters referring to the L1 data cache, there is another counter with the same contents referring to the L1 Instruction cache. The counters here have a numbered identification. The extended basic counter set contains the following cache data:

- | | |
|--------------------|---|
| Counter 129 | A directory write to the L1 data cache directory where the installed cache line was sourced from the L1.5 cache. |
| Counter 130 | A directory write to the L1 data cache directory where the installed cache line was sourced from the L2 cache that is on the same book as the Instruction cache. |
| Counter 133 | A directory write to the Level 1 data cache directory where the installed cache line was sourced from an L2 cache that is not on the same book as the data cache (Remote L2 cache). |
| Counter 134 | A directory write to the Level 1 data cache where the installed cache line was sourced from memory that is attached to the same book as the data cache (Local Memory).

Directory writes that are sourced from memory that is attached to a different book than the book where the Level 1 cache is (Remote Memory) can be back-calculated by using Counter 2 Level 1 I-Cache Directory Write Count and Counter 4 Level 1 D-Cache Directory Write Count that are part of the basic counter set in the CPU-Measurement facility. |
| Counter 136 | A directory write to the L1 data cache where the line was originally in a shared state in the cache but has been updated to be in the Exclusive state that allows stores to the cache line. |
| Counter 137 | A cache line in the Level 1 Instruction cache has been invalidated by a store on the same CPU as the Level 1 Instruction cache. |
| Counter 147 | Incremented by one for every store sent to Level 2 (L1.5) cache. |

With these counters as input, specific programs can quickly compute a performance estimation, such as average cache penalty, to determine whether performance problems exist when running an application or a piece of z/OS code.

Prefetch data, a new cache-directive z10 instruction, initiates the bring in and release of lines from and to the processor storage subsystem. Software perfecting instructions have been available in the industry for some time. These directives can be used by software (directed by either programmers or compilers) to attempt to maximize the number of cache hits and minimize the number of cycles spent waiting for data from main memory.

2.35 Virtual storage concept

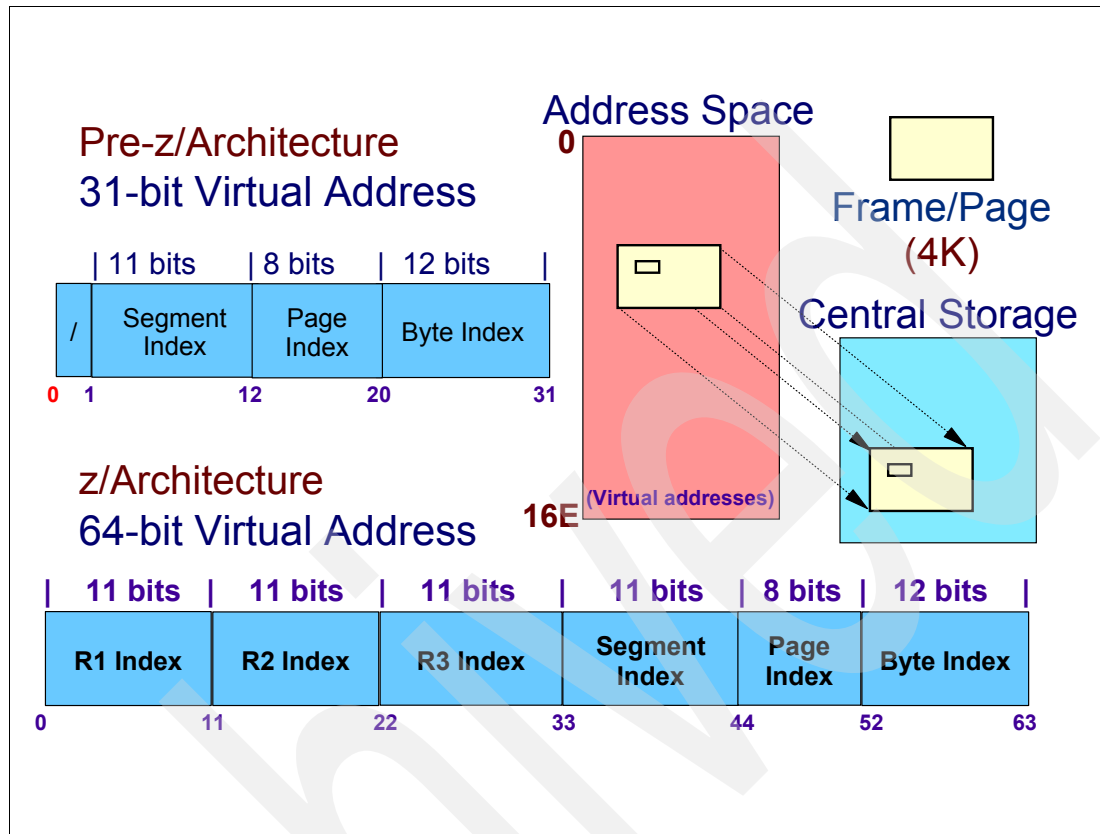


Figure 2-35 Virtual storage concept

Virtual storage concept

One piece of data captured by CPMF is the use of translation look-aside buffers. To understand the performance aspects of this data, we need to review the basic concepts of virtual storage.

With the implementation of virtual storage, all running programs can refer to many more addresses (for instruction and operands) than the number of bytes in main storage. The benefit of implementing virtual storage can become more apparent from the following scenario.

Scenario

Imagine that you have 1 GB available in main storage, and that all your programs have a hypothetical size of 100 MB. Without implementing virtual storage you are able to load only 10 programs that serve, perhaps, 10 concurrent business transactions. The drawback is that, for most of the time, the tasks running these programs are in wait due to ongoing I/O operations (commercial transactions are I/O-bound). Then the CP utilization is a small figure, and on top of that the queue for arriving transactions can grow, thereby deteriorating the overall response time.

With the implementation of virtual storage, only 20% of the referred program (if it is needed) stays loaded in main storage. The pieces of the program needed in main storage are named the working set. Thus, the main storage resource is much better utilized, together with the CP, because now the concurrent number of active (concurrent) transactions is much larger.

Consequences of implementing virtual storage

Note the following consequences of implementing virtual storage:

- ▶ The addresses referenced by the PU (named virtual addresses) during the program execution (to locate instructions and operands) are different from the ones that indicate the byte locations (known as real addresses). This implies the use of a translator.
- ▶ The contents of programs and operands that are not used much and which do not fit (overflow) in main storage are kept in DASD. In z/OS, these data sets are known as page data sets.

To implement virtual storage, the following design topics are implemented:

- ▶ main storage is formatted in 4 KB chunks known as frames.
- ▶ DASD space is formatted in page data sets with 4 KB blocks known as slots.
- ▶ The address space (the set of all virtual addresses that a program can reference) is formatted in blocks of 4 K addresses known as pages. The z10 EC server, optionally a page, can have a size of 1 M addresses (large pages). However, this is only possible for pages containing addresses larger than 2 G.
- ▶ A set of 256 contiguous pages (1 M addresses) is known as a segment.
- ▶ A set of 2 K contiguous segments (2 Giga addresses) is known as a third region.
- ▶ A set of 2 K contiguous third regions (4 Tera addresses) is known as a second region.
- ▶ A set of 2 K contiguous second regions (8 Peta addresses) is known as a first region.

There are 2 K first regions. With $(2 K * 8\text{-Peta})$, we reach the amount of 16-Exa addresses.

Figure 2-35 shows how you can fragment a virtual address referred by the PU, when in z/Architecture (64-bit):

- ▶ The first 11 bits indicate the first region index in the address space.
- ▶ The second 11 bits indicate the second region index in a first region.
- ▶ The third 11 bits indicate the third region index in a second region.
- ▶ The fourth 11 bits indicate the segment index in a third region.
- ▶ The next 4 bits indicate the page index in a segment.
- ▶ The last 12 bits indicate the byte index in a page.

2.36 Program status word (PSW) format

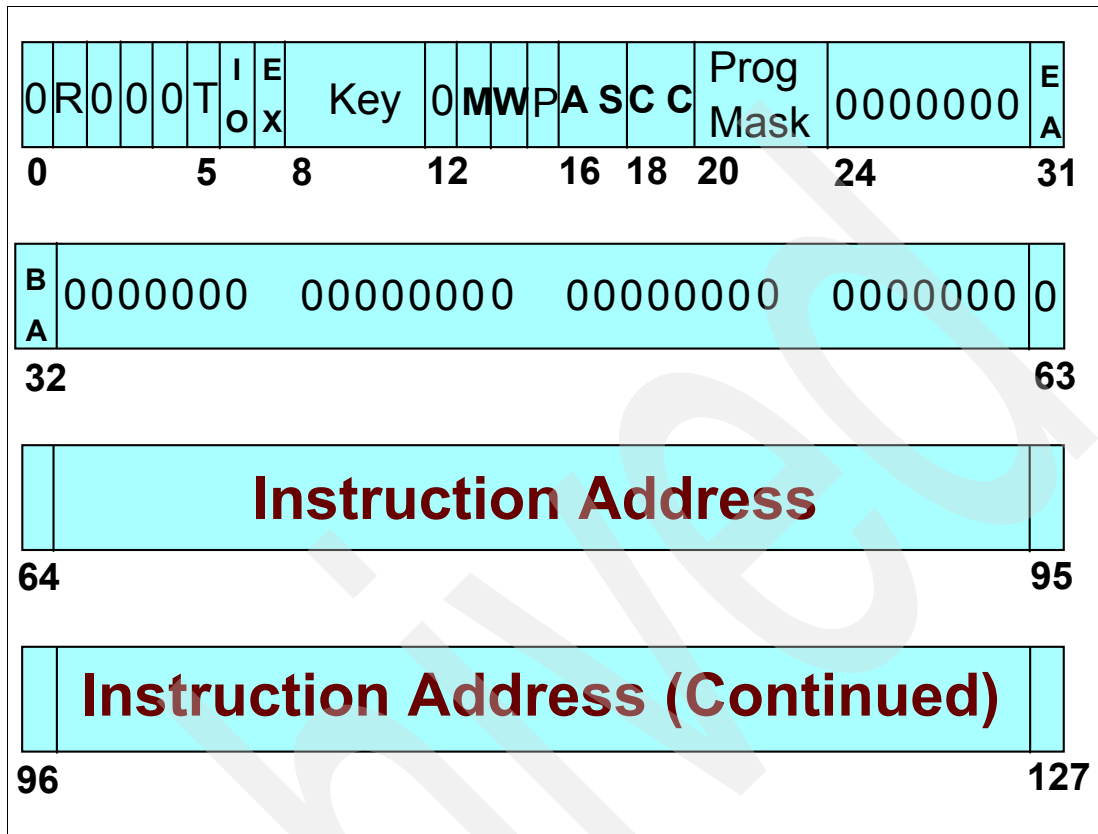


Figure 2-36 Program status word (PSW) format

PER mask - R (bit 1)

Bit 1 controls whether the CP is enabled for interrupts associated with program event recording (PER). When the bit is zero (0), no PER event can cause an interruption. When the bit is one, interruptions are permitted, subject to the PER event mask bits in control register 9.

DAT mode - T (bit 5)

Bit 5 controls whether implicit dynamic address translation of logical and instruction addresses used to access storage takes place. When the bit is zero, DAT is off, and logical and instruction addresses are treated as real addresses. When the bit is one, DAT is on, and the dynamic address translation mechanism is invoked.

I/O mask - IO (bit 6)

Bit 6 controls whether the CP is enabled for I/O interruptions. When the bit is zero, an I/O interruption cannot occur. When the bit is one, I/O interruptions are subject to the I/O interruption subclass mask bits in control register 6. When an I/O interruption subclass mask bit is zero, an I/O interruption for that I/O interruption subclass cannot occur. When the I/O interruption subclass mask bit is one, an I/O interruption for that I/O interruption subclass can occur.

External mask - EX (bit 7)

Bit 7 controls whether the CP is enabled for interruption by conditions included in the external class. When the bit is zero, an external interruption cannot occur. When the bit is one, an

external interruption is subject to the corresponding external subclass mask bits in control register 0. When the subclass-mask bit is zero, conditions associated with the subclass cannot cause an interruption. When the subclass-mask bit is one, an interruption in that subclass can occur.

PSW key (bits 8-11)

Bits 8-11 form the access key for storage references by the CP. If the reference is subject to key-controlled protection, the PSW key is matched with a storage key when information is stored or when information is fetched from a location that is protected against fetching. However, for one of the operands of each of MOVE TO PRIMARY, MOVE TO SECONDARY, MOVE WITH KEY, MOVE WITH SOURCE KEY, and MOVE WITH DESTINATION KEY, an access key specified as an operand is used instead of the PSW key.

Processor check mask - M (bit 13)

Bit 13 controls whether the CP is enabled for interruption by processor check conditions. When the bit is zero, a processor check interruption cannot occur. When the bit is one, processor check interruptions due to system damage and instruction-processing damage are permitted, but interruptions due to other processor check subclass conditions are subject to the subclass mask bits in control register 14.

Wait state - W (bit 14)

When bit 14 is one, the CP is waiting; that is, no instructions are processed by the CP but interruptions can occur. When bit 14 is zero, instruction fetching and execution occur in the normal manner. The wait indicator is on when the bit is one. When in wait state, the only way of getting out of such state is through an interruption.

Certain bits, when off in the current PSW, place the CP in a disabled state; the CP does not accept Interrupts. So when z/OS, for any error reason (software or hardware) decides to stop a CP, it sets the PSW to the Disable and Wait state, forcing an IPL to restore the CP to the running state.

Problem state - P (bit 15)

When bit 15 is one, the CP is in the problem state. When bit 15 is zero, the CP is in the supervisor state. In the supervisor state, *all* instructions are valid.

In the problem state, the only instructions that are valid are those which provide meaningful information to the problem program and that cannot affect system integrity; such instructions are known as unprivileged instructions.

The instructions that are never valid in the problem state are called privileged instructions. When a CP in the problem state attempts to execute a privileged instruction, a privileged operation exception is recognized. Another group of instructions, called semi-privileged instructions, are executed by a CP in the problem state only if specific authority tests are met; otherwise, a privileged operation exception or a special operation exception is recognized.

Address space control -AS (bits 16-17)

Bits 16 and 17, in conjunction with PSW bit 5, control the translation mode.

Condition code - CC (bits 18-19)

Bits 18 and 19 are the two bits of the condition code. The condition code is set to 0, 1, 2, or 3, depending on the result obtained in executing certain instructions. Most arithmetic and logical operations, and some other operations, set the condition code. The instruction BRANCH ON

CONDITION can specify any selection of the condition-code values as a criterion for branching.

The part of the CP that executes instructions is called the arithmetic logic unit (ALU). The ALU has four bits internally that are set by certain instructions. At the end of these instructions, this 4-bit configuration is mapped into bits 18 and 19 of the current PSW.

As an example, the instruction COMPARE establishes a comparison between two operands. The result of the comparison is placed in the CC of the current PSW, as follows:

- ▶ If CC=00, then the operands are equal.
- ▶ If CC=01, then first operand is lower.
- ▶ If CC=10, then first operand is greater.

To test the contents of a CC (set by a previous instruction), use the BRANCH ON CONDITION (BC) instruction. It contains an address of another instruction (branch address) to be executed, depending on the comparison of the CC and a mask M. The instruction address in the current PSW is replaced by the branch address if the condition code has one of the values specified by M; otherwise, normal instruction sequencing proceeds with the normal updated instruction address. The types of codes are listed here:

- ▶ Condition code (bits 18,19 PSW)
- ▶ Return code - a code associated with how a program ended
- ▶ Completion code - a code associated with how a task ended
- ▶ Reason code - a code passed in the GPR 15 detailing how a task ended

Program mask (bits 20-23)

During the execution of an arithmetic instruction the CP can find unusual (or error) conditions such as overflows, lost of significance, or underflow. In such cases, the CP generates a program interrupt.

When this interrupt is treated by z/OS, usually the current task is abnormally ended (abend). However, in certain situations programmers do not want an abend, so by using the instruction SET PROGRAM MASK (SPM), they can mask such interrupts by setting some of the program mask bits to OFF. Each bit is associated with one type of condition:

- ▶ Fixed point overflow (bit 20)
- ▶ Decimal overflow (bit 21)
- ▶ Exponent underflow (bit 22)
- ▶ Significance (bit 23)

The active program is informed about these events through the condition code posted by the instruction where the events described happened.

The contents of the CP can be totally changed by two events:

- ▶ By loading a new PSW from storage after an interruption
- ▶ By executing the instruction LPSW, which copies 128 bits from memory to the current PSW

Extended addressing mode - EA, BA (bits 31- 32)

The combination of bits 31 and 32 identifies the addressing mode (24, 31, or 64) of the running program. Bit 31 controls the size of effective addresses and effective address generation in conjunction with bit 32, which is the basic addressing mode bit. When bit 31 is zero, the addressing mode is controlled by bit 32. When bits 31 and 32 are both one, 64-bit addressing is specified.

2.37 Dynamic address translation (DAT)

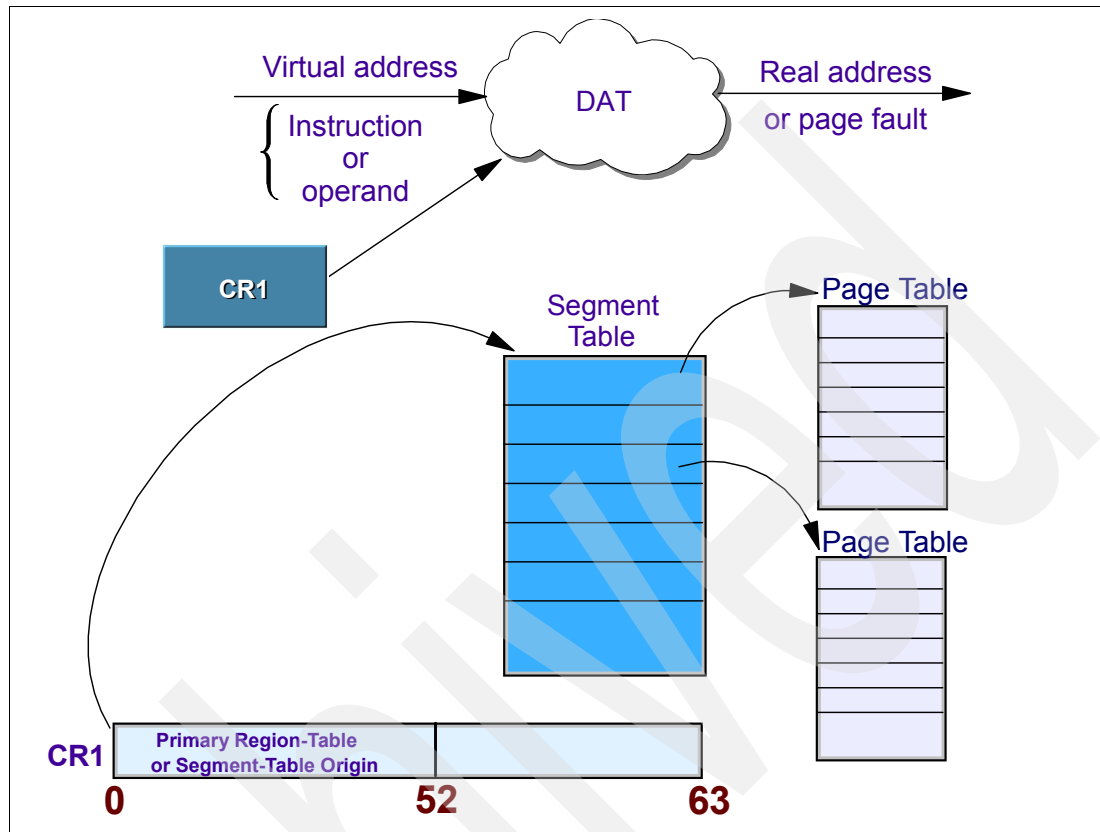


Figure 2-37 Dynamic address translation

Dynamic address translation

Dynamic address translation (DAT) is the name of the hardware within any PU in charge of translating a virtual address (instruction or operand) during a PU CS reference into the corresponding real address. If such a translation is not possible, DAT produces a page fault (a type of program interrupt). Performance-wise, the translation must be done through fast hardware because for each instruction, on average, the DAT is invoked twice.

Control registers

The CPU has 16 control registers, each having 64-bit positions. The bit positions in the registers are assigned to particular facilities in the system, such as program event recording, and are used either to specify that an operation can take place or to furnish special information required by the facility.

A control register (CR) is identified by the numbers 0-15 and is designated by 4-bit R fields in the instructions LOAD CONTROL and STORE CONTROL. Multiple control registers can be addressed by these instructions.

PSW bit 5 (DAT)

DAT is active only when bit 5 in the current PSW is one. This bit is set by z/OS and when it is OFF (which seldom occurs), it means that the virtual address is equal to the real address. To translate a virtual address into a real address, DAT uses hierarchical indexed tables managed by z/OS in main storage.

The first hierarchical table is pointed to by control register (CR) 1, as shown in Figure 2-37. The designation (origin and length) of the highest-level table for a specific address space is called an address space control element.

Each entry in a segment table points to a 256 page table. Each page table entry has the real address of the main storage frame that has the contents (instruction or operands) pointed to by the address in that page.

Dynamic address translation overview

Dynamic address translation is the process of translating a virtual address during a storage reference into the corresponding real address. A segment table designation or region table designation causes translation to be performed by means of tables established by the operating system in real or absolute storage.

z/OS translation tables

There are three levels of translating tables: segment table, region table, and page table. Each entry in the segment table points to a page table. Each entry in the page table points to the location of the frame associated with that page.

In the process of translation when using a segment table designation or a region table designation, three types of units of information are recognized:

- Region** This refers to a block of sequential virtual addresses spanning 2 Gbytes and beginning at the 4 Gbyte boundary (above the bar).
- Segment** This refers to a block of sequential virtual addresses spanning 1 MB and beginning at a 1 MB boundary.
- Page** This refers to a block of sequential virtual addresses spanning 4 KB and beginning at a 4 KB boundary. Starting with the z10 EC server, optionally a page can have a size of 1 M addresses (large pages). However, this is only possible for pages above the bar. All those pages are fixed in main storage.

2.38 Translating a 31-bit virtual address

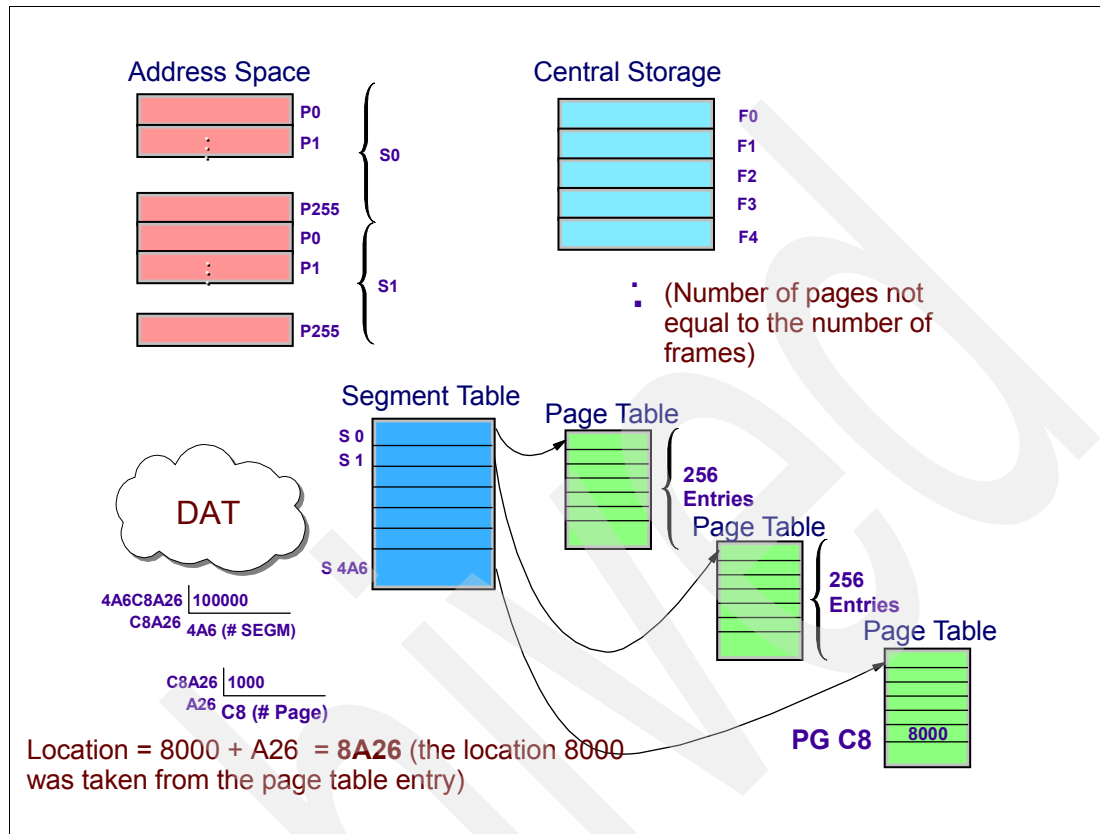


Figure 2-38 Translating a 31-bit virtual address

Translating a 31-bit virtual address

The example shown in Figure 2-38 on page 123 illustrates the DAT translation of virtual address X'4A6C8A26', as explained here:

- ▶ Receive the address from the PU (it can be the address of an operand or an instruction).
- ▶ Calculate the segment index, the page index, and the byte index as described in "Virtual storage concept" on page 116:
 - Divide the address by 1M (segment size, that is X'100000'). The quotient is the segment index (4A6) and the rest (C8A26) is the displacement in the segment 4A6.
 - Divide the displacement by 4 K (page size, that is, X'1000'). The quotient is the page index (C8) and the rest is the byte index (A36).
- ▶ Locate the segment table through the contents of CR 1.
- ▶ Look for the segment entry corresponding to segment 4A6.
- ▶ Locate the page table pointed by such segment entry.
- ▶ Look for the page entry corresponding to page C8.
- ▶ In Figure 2-38 on page 123, this entry contains the real address X'8000', which is the pointer for a CS frame.
- ▶ Add the byte index (A26) to the frame real address (8000) to produce the real address X'8A26'.
- ▶ Pass back this result to the PU to allow access to the memory contents.

2.39 Translating a 64-bit virtual address

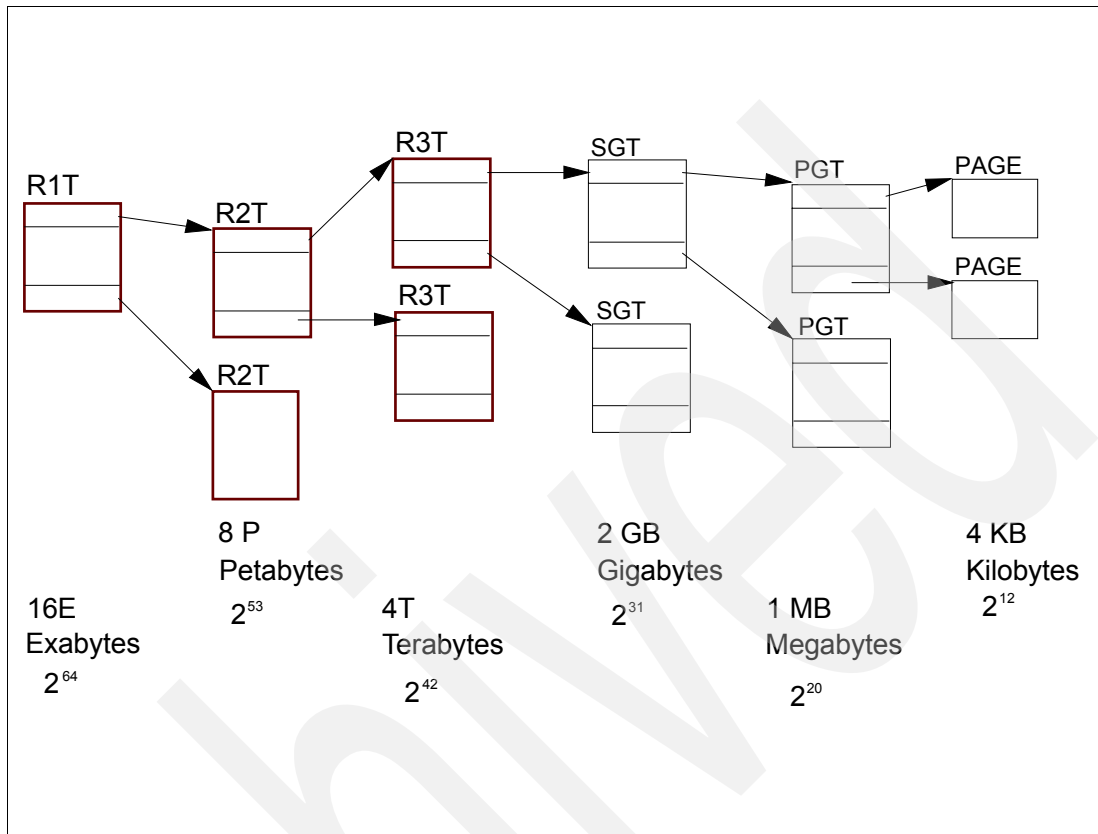


Figure 2-39 Translating a 64-bit virtual address

Translating a 64-bit virtual address (up to 16 Exa)

The logic here is equivalent to that of translating a 31-bit address; see 2.38, “Translating a 31-bit virtual address” on page 123 for more information. The major difference is that there are three additional levels of translation tables, called *region tables*. They are called third region table (R3T), the second region table (R2T), and the first region table (R1T). They map the regions of an address space. Note the following points:

- ▶ A third region has 2 G addresses.
- ▶ A second region has 4 T addresses ($2K * 2G$).
- ▶ A first region has 8 Peta addresses ($2K * 4T$).

Region tables

The region tables are 16 KB in length, and there are 2048 (2 K) entries per table. Each table maps the regions described in 2.35, “Virtual storage concept” on page 116.

When z/OS creates an address space, it has 2 G addresses. In this case there is only one segment table and page tables to map the virtual address located in this address space. When an application (or z/OS itself) requires virtual storage above the bar (meaning above 2 G), z/OS creates the R3T. The R3T table has 2048 segment table pointers, and provides addressability to 4 T addresses.

When virtual storage greater than 4-T is needed, z/OS creates an R2T. An R2T has 2048 R3T table pointers and provides addressability to 8 P addresses.

An R1T is created when virtual storage is located above 8 P. The R1T has 2048 R2T table pointers, and provides addressability to 16 EB.

Region table creation

To summarize, z/OS only creates additional levels of region tables when it is necessary to back storage that is mapped. So for example, if an application requests 60 PB of virtual storage, the necessary R2T, R3T, segment table, and page tables are only created if they are needed to back a referenced page. Up to five lookup tables can be needed by DAT to do translation, but the translation only starts from the table that provides translation for the highest usable virtual address in the address space.

Figure 2-39 shows the page table hierarchy and the size of virtual storage each table covers. Segment tables and page table formats remain the same as for virtual addresses below the bar.

When translating a 64-bit virtual address, DAT is informed by the contents of CR1 about the nature (segment table or third region table or second region table or first region table) of the first hierarchical table pointed to by the register. After this identification is complete, the translation process is the same as that described previously.

2.40 Large page support

- ❑ Memory architecture supports large (1 MB) pages.
 - When large pages are used in addition to existing 4 KB page size, they are expected to reduce memory management overhead for exploiting applications.
- ❑ With the z10 EC models, page frames can be allocated with a 4 K size.
 - The z10 EC supports a larger page frame size of 1MB.
 - Long-running, memory access-intensive applications will benefit from large page frames.
 - Large pages are treated as fixed pages and are never paged out.

Figure 2-40 Large page support

Large page support

Large page usage is primarily intended to provide performance improvements to applications that allocate a significant amount of memory and frequently access that memory. This support is for pages that are 1 MB in size. These pages are in addition to the existing 4 KB pages. The size of pages and page frames has been 4 KB for a long time.

z10 EC models

The z10 EC server introduces the capability of having, in addition to pages with a size of 4 KB, large pages with a size of 1 MB. This is a performance item addressing particular workloads and relates to large main storage usage. Large page support is exclusive to System z10 servers. Note that both page frame sizes can be used simultaneously.

Application use

This support is primarily of benefit to long-running applications that are memory access-intensive. Large pages are not recommended for general use. Short-lived processes with small working sets are normally not good candidates for large pages and see little to no improvement. The use of large pages must be decided based on knowledge obtained from measurement of memory usage and page translation overhead for a specific workload.

Large pages are treated as fixed pages and are never paged out. They are only available for 64-bit virtual private storage such as virtual memory located above the 2 GB bar.

2.41 Translation lookaside buffer (TLB)

- ❑ DAT is hardware contained in each PU in charge of translating virtual addresses into real addresses.
 - During such translation, DAT may access up to 5 CS tables (third region, second region, first region, segment and page) managed by z/OS.
- ❑ To avoid DAT accesses to CS, TLBs were introduced.
 - TLBs are a fast array memory within each PU.
 - After translating using CS tables, DAT keeps the relation page/frame in a TLB entry and the next time before going through a CS table translation, DAT inspects TLBs looking for the referred page.
 - In case of a hit, the translation process is much faster.

Figure 2-41 Translation lookaside buffer (TLB)

Dynamic address translation

As previously mentioned, dynamic address translation (DAT) is the process of translating a virtual address during a storage reference into the corresponding real address. If the virtual address is already in main storage (CS), the DAT process can be accelerated through the use of a translation lookaside buffer. If the virtual address is not in main storage, a page fault interrupt occurs and z/OS is notified and brings the page in from auxiliary storage.

Looking at this process more closely reveals that the machine can present any one of a number of different types of faults. A type, region, segment, or page fault will be presented depending on at which point in the DAT structure invalid entries are found. The faults repeat down the DAT structure until ultimately a page fault is presented and the virtual page is brought into main storage either for the first time (there is no copy on auxiliary storage) or by bringing the page in from auxiliary storage.

DAT is the piece of hardware contained in each PU in charge of translating virtual addresses into real addresses. This task is accomplished through the access of main storage (CS) tables managed by z/OS. With 64-bit addressing it is possible to have up to five tables involved in such translation (third region, second region, first region, segment, and page). To make DAT performance feasible, avoid these CS accesses.

Translation look-aside buffer (TLB)

DAT is implemented by both hardware and software through the use of page tables, segment tables, region tables, and translation lookaside buffers. DAT allows different address spaces

to share the same program or other data that is for read only. This is because virtual addresses in different address spaces can be made to translate to the same frame of main storage. Otherwise, there have to be many copies of the program or data, one for each address space. TLBs are located in a fast PU internal memory.

After going through a translation using CS tables, DAT keeps the relation page/frame in a TLB entry. Then, for the next translation before going through a CS table translation, DAT first inspects TLBs looking for the needed page. In case of a hit, the translation process is much faster.

Cache and TLBs

A cache can hold translation lookaside buffers (TLBs), which contain the mapping from virtual address to real address of recently used pages of instruction text or data. Depending on the processor architecture and model, processors have from one to several caches to hold the following:

- ▶ Parts of executing programs
- ▶ Data used by executing programs
- ▶ TLBs

If a cache miss occurs, loading a complete cache line can take dozens of processor cycles. If a TLB miss occurs, calculating the virtual-to-real mapping of a page can take several dozen cycles. The exact cost is implementation-dependent.

2.42 TLB performance and 1 MB pages

- ❑ z10 EC has 512 TLB entries per PU.
- ❑ Due to the explosion of virtual addresses because of 64-bit capability, 512 entries is not enough to generate perfect performance (a high TLB hit ratio).
- ❑ The introduction of 1 MB pages decreases the number of pages in virtual storage (fewer larger pages instead of many small pages), thus improving DAT performance.
- ❑ To avoid the reformatting of page data sets and high data rate during ASM paging operations, those pages must be fixed in central storage (CS).

Figure 2-42 TLB performance

TLB performance

A PU is the generic term for the z/Architecture processor on the multi-chip module (MCM). A PU is imbedded in a System z chip core. Each PU is a superscalar processor. Each L1 cache has a translation lookaside buffer (TLB) of 512 entries associated with it. In addition, a secondary TLB is used to further enhance performance. This structure supports large working sets, multiple address spaces, and a two-level virtualization architecture.

Large pages cause the translation lookaside buffer (TLB) to better represent the working set and suffer fewer misses by allowing a single TLB entry to cover more address translations. Exploiters of large pages are better represented in the TLB and are expected to perform better. However, due to the explosion of virtual addresses because of the 64-bit capability, 512 is not enough to generate desirable performance.

IBM developed two actions to address this issue:

- ▶ Allowing 1 MB pages. This decreases the total population of pages, to improve DAT performance.
- ▶ Allowing CPMF to track TLBs effectiveness.

There are two types of TLBs, namely TLB1 and TLB2. Only TLB2s are candidates to contain 1 MB pages of information. Then, for every instruction we can save twice (on average) going to CS (or cache) to find the tables used to translate the virtual addresses to real addresses.

This support primarily benefits long-running applications that are memory access-intensive. Large pages are not recommended for general use. Short-lived processes with small working sets are normally not good candidates for large pages and see little to no improvement. The use of large pages must be decided based on knowledge obtained from measurement of memory usage and page translation overhead for a specific workload.

Large pages are fixed

Large pages are treated as fixed pages and are never paged out. They are only available for 64-bit virtual private storage such as virtual memory located above 2 GB.

2.43 Large page support specifications

- ❑ Memory reserved for large page support can be defined with a new parameter in the IEASYSxx parmlib member:
 - LFAREA=xx% | xxxxxxM | xxxxxxG
 - xx% - this indicates that the amount of real storage to be used for large pages is specified as a percentage of all online real storage available at IPL time. The maximum amount of real storage that can be used to back large pages is 80% of the online storage available at IPL minus 2 GB.
 - xxxxxxM - Real storage at IPL in megabytes.
 - xxxxxxG - Real storage at IPL in gigabytes.

Figure 2-43 Specify large page support in the IEASYSxx parmlib member

IEASYSxx parmlib member

These large pages consume real storage and are non-pageable. System programmers must carefully consider which applications are granted access to large pages. Not all applications benefit from using large pages. Long-running memory-intensive applications benefit most from using large pages. Short-lived processes with a small memory working set are not good candidates. System programmers define the amount of real storage that can be used for large pages with the LFAREA system parameter in the IEASYSxx parmlib member. The key factors to consider when you grant access to large pages are:

- ▶ Memory usage
- ▶ Page translation overhead of the workload
- ▶ Available large frame area

LFAREA specification

The LFAREA=(xxxxxx) parameter specifies the amount of real storage to be made available for 1 MB pages. The value specified on this parameter indicates the amount of online real storage at IPL that is to be used to back large pages, as follows:

- ▶ xxxxxxM or xxxxM

This indicates the amount of online real storage at IPL time, in megabytes, that is to be used to back large pages. The maximum amount of real storage that can be used to back large pages is 80% of the online storage available at IPL minus 2 GB. If the amount

specified exceeds this value, the specification is rejected and the system issues a prompt for a new value to be specified on the LFAREA parameter.

▶ xxxxxG or xxxxG

This indicates the amount of online real storage available at IPL time, in gigabytes, that is to be used for large pages. The maximum amount of real storage that can be used to back large pages is 80% of the online storage available at IPL minus 2 GB. If the amount specified exceeds this value, the specification is rejected and the system issues a prompt for a new value to be specified on the LFAREA parameter.

▶ xx%

This indicates that the amount of real storage to be used for large pages is specified as a percentage of all online real storage available at IPL time. The maximum amount of real storage that can be used to back large pages is 80% of the online storage available at IPL minus 2 GB. If the amount specified exceeds this value, the specification is rejected and the system issues a prompt for a new value to be specified on the LFAREA parameter.

Note: The maximum amount of real storage that can be used to back large pages is 80% of the online storage available at IPL minus 2 GB. The formula to calculate the maximum amount of real storage that can be used to back large pages is as follows:

▶ For calculations in gigabytes:

$$x = \text{Online Real Storage at IPL in G}$$
$$\text{MAX_LFAREA} = (x - 2G) * .8$$

▶ For calculations in megabytes:

$$y = \text{Online Real Storage at IPL in M}$$
$$\text{MAX_LFAREA} = (y - 2048M) * .8$$

2.44 CPMF extended counters on TLB

- ❑ Counter 138 – A translation entry has been written into the Level 1 Instruction TLB1.
- ❑ Counter 139 – A translation entry has been written to the Level 1 Data TLB1.
- ❑ Counter 140 – A translation entry has been written to the Level 2 TLB Page Table Entry arrays.
- ❑ Counter 141 – A translation entry has been written to the Level 2 TLB Common Region Segment Table Entry arrays.
- ❑ Counter 142 – A translation entry has been written to the Level 2 TLB Common Region Segment Table Entry arrays for a one-megabyte large page translation.
- ❑ Counter 145 – Level 1 Instruction TLB miss in progress. Incremented by one for every cycle an ITLB1 miss is in progress.
- ❑ Counter 146 – Level 1 Data TLB miss in progress. Incremented by one for every cycle that a DTLB1 miss is in progress.

Figure 2-44 CPMF extended counters on TLB performance

CPMF extended counters on TLB performance

With the information contained in such counters we can evaluate the real TLB contribution to the general PU performance and also to pinpoint bottlenecks. The extended counter set contains the following cache data:

- ▶ Counter 138 - A translation entry has been written into the Level 1 instruction translation lookaside buffer (ITLB1).
- ▶ Counter 139 - A translation entry has been written to the Level 1 data translation lookaside buffer (DTLB1).
- ▶ Counter 140 - A translation entry has been written to the Level 2 TLB Page Table Entry arrays.
- ▶ Counter 141 - A translation entry has been written to the Level 2 TLB common region segment table entry arrays.
- ▶ Counter 142 - A translation entry has been written to the Level 2 TLB common region segment table entry arrays for a one-megabyte large page translation.
- ▶ Counter 145 - Level 1 instruction TLB miss in progress. Incremented by one for every cycle an ITLB1 miss is in progress.
- ▶ Counter 146 - Level 1 data TLB miss in progress. Incremented by one for every cycle a DTLB1 miss is in progress.

2.45 IBM common cryptographic architecture (CCA)

- ❑ Key management, which includes generation and exchange of keys securely across networks and between application programs
- ❑ Data integrity, with the use of a Message Authentication Code (MAC), Modification Detection Code (MDC), or digital signature
- ❑ Data confidentiality, with the use of encryption and decryption capabilities accessible at all levels of a network protocol stack
- ❑ Personal authentication, with PIN generation, verification, and translation

Figure 2-45 Common cryptographic architecture

Common cryptographic architecture

The IBM Common Cryptographic Architecture (CCA) defines a set of cryptographic functions, external interfaces, and a set of key management rules which pertain both to the data encryption standard (DES)-based symmetric algorithms and the Public Key Algorithm (PKA) asymmetric algorithms. These provide a consistent, end-to-end, cryptographic architecture across several IBM platforms, such as OS/390, AIX®, OS/400®, OS/2, and Windows® NT, which conforms to American and International Standards.

Functions of the Common Cryptographic Architecture define services for the following areas.

Key management

Key management, which includes the generation and exchange of keys securely across networks and between application programs. The exchanged key is securely encrypted using either DES or a Public Key Algorithm used in the context of symmetric key management.

Data integrity

Data integrity, with the use of a message authentication code (MAC), modification detection code (MDC), or digital signature, as follows:

- ▶ Message authentication is the process of verifying the integrity of transmitted messages. Message authentication code (MAC) processing enables you to verify that a message has not been altered. You can use a MAC to check that a message you receive is the same

message that the message originator sent. The message itself may be in clear or encrypted form. MAC keys are either single-length (64-bit) or double-length (128-bit) keys.

- ▶ An alternative approach to data-integrity checking uses a standard key value and multiple iterations of the DES algorithm to generate a modification detection code (MDC). In this approach to data-integrity checking, the MDC must be received from a trusted source. The person who wants to authenticate the data recomputes the MDC and compares the result with the MDC that was sent with the data.
- ▶ Public key cryptographic algorithms are used in processes that simplify the distribution of secret keys, assuring data integrity and provide nonrepudiation through the use of digital signatures.

Data confidentiality

Data confidentiality, with the use of encryption and decryption capabilities, is accessible at all levels of a network protocol stack. The Commercial Data Masking Facility (CDMF) defines a scrambling technique for data confidentiality. CDMF is a substitute for DES for clients that were previously prohibited from receiving IBM products that support DES data confidentiality services.

The CDMF data confidentiality algorithm is a cryptographic system that provides data masking and unmasking. The algorithm includes both a key-shortening process and a standard DES encryption and decryption process. The key-shortening process shortens the key to an effective length of 40 bits prior to its use in the data masking process. CDMF uses the DES algorithm with the shortened key to ensure confidence in the CDMF algorithm.

Personal authentication

Personal authentication, with PIN generation, is used for verification and translation. Personal authentication is the process of validating personal identities in a financial transaction system. The personal identification number (PIN) is the basis for verifying the identity of a customer across the financial industry networks. A PIN is a number that the bank customer enters into an automated teller machine (ATM) to identify and validate a request for an ATM service.

2.46 Symmetric cryptography

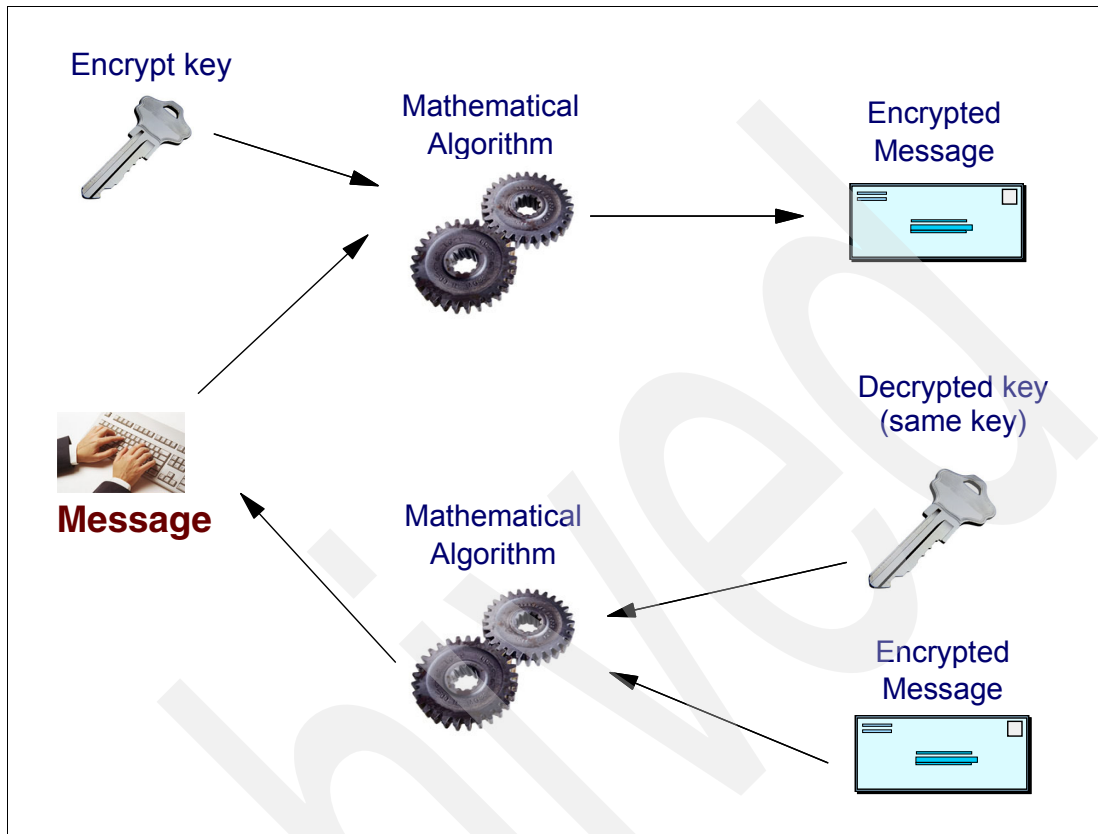


Figure 2-46 Symmetric cryptography

Symmetric cryptography

ICSF supports several symmetric cryptography algorithms, as follows:

- ▶ The Data Encryption Algorithm

For commercial business applications, the cryptographic process that is known as the Data Encryption Algorithm (DEA) (1) has been widely adopted. The Data Encryption Standard (DES), as well as other documents, defines how to use the DES algorithm to encipher data. The Data Encryption Standard is the basis for many other processes for concealing data, such as protection of passwords and personal identification numbers (PINs).

DES uses a key to vary the way that the algorithm processes the data. DES data-encrypting keys can be single-, double-, or triple-length. A single-length DES key is a 56-bit piece of data that is normally retained in 8 bytes of data. Each eighth bit of the key data is designated as a parity bit. A symmetric cryptographic system uses the same key both to transform the original data (plaintext) to its disguised, enciphered form (ciphertext) and to return it to its plain text form.

- ▶ The Advanced Encryption Standard

ICSF supports the Advanced Encryption Standard algorithm for data privacy. This provides strong encryption. Key lengths of 128-bits, 192-bits and 256-bits are supported. The algorithm has the same availability as triple DES. Secure key AES is available if running on IBM System z10 Enterprise Class and IBM System z10 Business Class with the Nov. 2008 or later licensed internal code (LIC).

- ▶ Commercial Data Masking Facility

The Commercial Data Masking Facility (CDMF) defines a scrambling technique for data confidentiality. CDMF is a substitute for DES for clients that were previously prohibited from receiving IBM products that support DES data confidentiality services.

Restriction: CDMF is only supported on the IBM eServer zSeries 800 and the IBM eServer zSeries 900.

The CDMF data confidentiality algorithm is a cryptographic system that provides data masking and unmasking. The algorithm includes both a key-shortening process and a standard DES encryption and decryption process. The first process shortens the key to an effective length of 40 bits prior to its use in the data masking process. CDMF uses the DES algorithm with the shortened key to ensure confidence in the CDMF algorithm.

2.47 Asymmetric cryptography

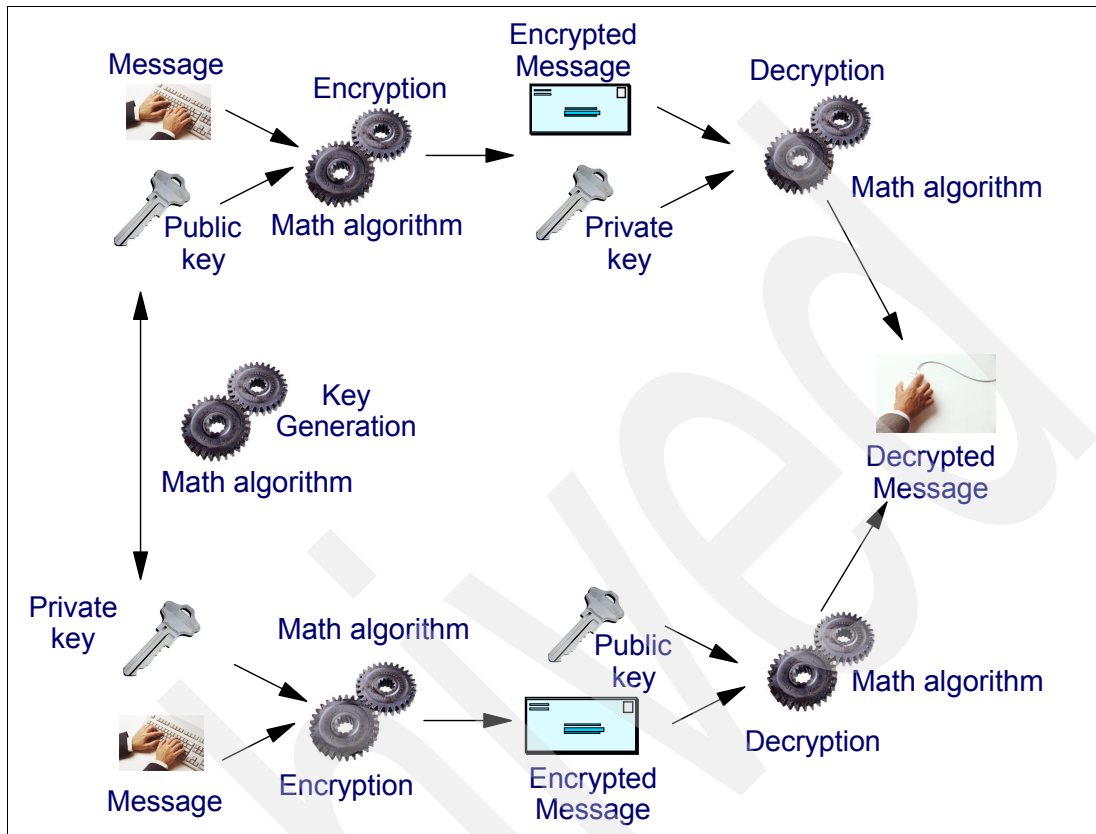


Figure 2-47 Asymmetric cryptography

Asymmetric cryptography

In an asymmetric cryptographic process one key is used to encipher the data, and a different but corresponding key is used to decipher the data. A system that uses this type of process is known as a public key system. The key that is used to encipher the data is widely known, but the corresponding key for deciphering the data is a secret. For example, many people can use your public key to send enciphered data to you with confidence, but only you should possess the secret key for deciphering the data.

Public key cryptographic algorithms are used in processes that simplify the distribution of secret keys, assuring data integrity and providing nonrepudiation through the use of digital signatures.

The widely known and tested public key algorithms use a relatively large key. The resulting computer processing time makes them less than ideal for data encryption that requires a high transaction rate. Public key systems, therefore, are often restricted to situations in which the characteristics of the public key algorithms have special value, such as digital signatures or key distribution. PKA calculation rates are fast enough to enable the common use of digital signatures.

ICSF supports these public key algorithms:

- ▶ Rivest-Shamir-Adelman (RSA).
- ▶ Digital Signature Standard (DSS) - DSS is only supported on the IBM eServer zSeries 800 and the IBM eServer zSeries 900.

2.48 Clear key and non-clear key algorithms

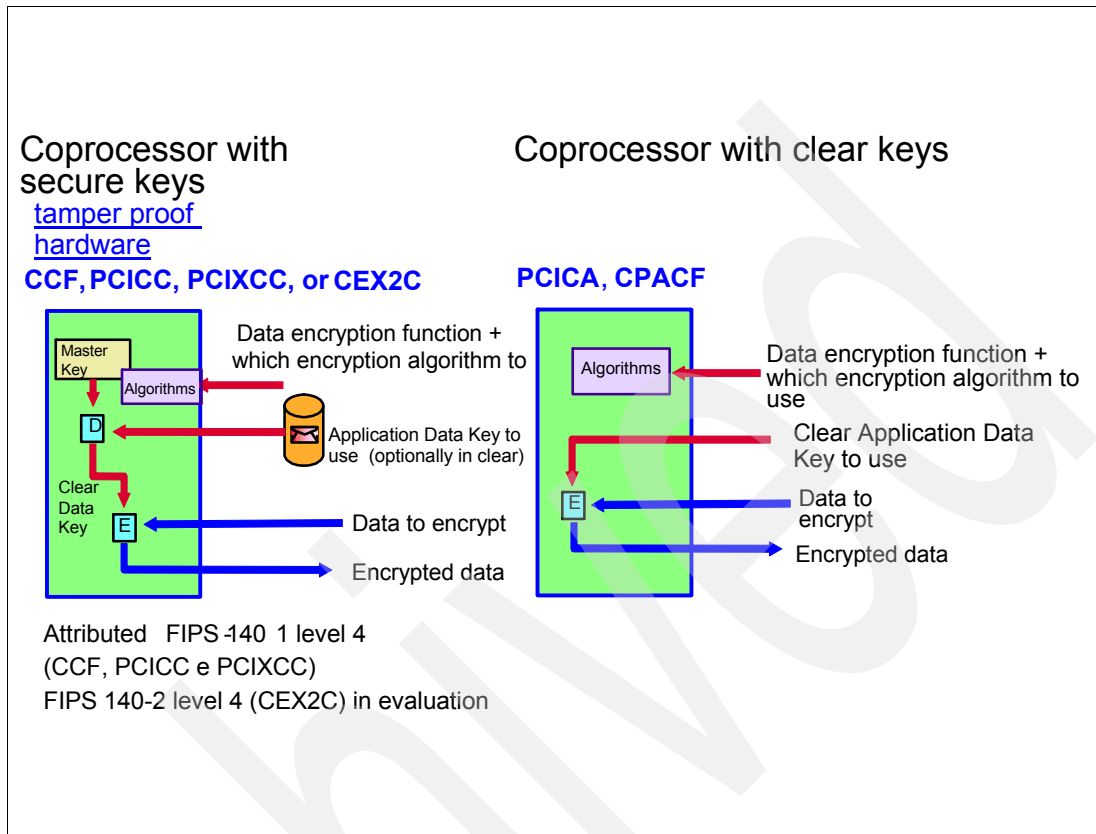


Figure 2-48 Clear and non-clear key algorithms

Cryptographic Coprocessor Feature

The Cryptographic Coprocessor Feature (CCF) can have up to two cryptographic coprocessors as high-speed extensions of the central processor. Each CCF contains both DES and PKA cryptographic processing units. You can configure the processor complex to run in either single-image mode or logical partition mode. If the CCF is in single-image mode, the same master keys must be installed on both CCFs. If you bring a second coprocessor online, ICSF verifies that the master keys are the same. If the DES master keys are different, ICSF will not use the second coprocessor. The PKA master keys must be the same on both Coprocessors in order to enable the PKA services. This feature is currently certified for Federal Information Processing Standard (FIPS) 140-1 level 4. This includes algorithmic certification under FIPS 46-2 (DES), FIPS 180-1 (Secure Hash Standard), and FIPS 186 (Digital Signature Standard).

Clear key and non-clear key algorithms

Clear key algorithms allow the master key be stored in z/OS central storage. The Crypto Express2 co-processor enables the user to:

- ▶ Encrypt and decrypt data utilizing secret-key (non-clear key) algorithms. Triple-length key DES and double-length key DES algorithms are supported.
- ▶ Generate, install, and distribute cryptographic keys securely, using both public and secret key cryptographic methods.

Crypto Express-2 has granted an APS 140-2 level 4 from the U.S. Government. CPACF only allows clear key algorithms.

2.49 Hardware crypto in a z10

- ❑ CP Assist for Cryptographic Functions (CPACF) in the PU chip - CP Synchronous
- ❑ Crypto Express2 (two adapters) in the I/O Cage - CP Asynchronous
- ❑ TKE workstation
 - TKE smart card reader

Figure 2-49 Hardware crypto in a z10

Hardware cryptography in a z10 machine

Cryptography in a z10 machine is implemented through:

- ▶ CP Assist for Cryptographic Functions (CPACF) in the PU chip - CP Synchronous
- ▶ Crypto Express2 (two adapters) in the I/O Cage - CP Asynchronous
- ▶ A TKE workstation, which may contain a TKE smart card reader.

CPACF

CPACF is an inbound synchronous crypto coprocessor located at the multichip module (MCM) in a book. It is “synchronous” because when it is working, the CPU that invoked it is in a microcode loop waiting synchronously by the ending of the CPACF operation. It is activated by the execution of several CPU zArchitecture instructions such as Cypher. It only executes clear key algorithms. CPMF counters apply to CPACF execution only.

Zooming into the CPACF, notice that it is a cryptographic coprocessor group consisting of a cipher coprocessor for the Data Encryption Standard (DES) and the Advanced Encryption Standard (AES) and a hash coprocessor for the Secure Hash Algorithm. Both coprocessors can perform cryptographic operations simultaneously.

Crypto Express-2

Crypto Express-2 is an outbound crypto coprocessor located in an I/O card in an I/O cage. It is asynchronous because the CPU does not wait for its processing. CPMF counters do not

apply to Crypto Express-2 activities, because performance monitors through z/OS can access performance data covering Crypto Express-2 functions.

TKE workstation

The Trusted Key Entry (TKE) workstation (Version 4.0 or higher) is available on the IBM eServer zSeries 990 IBM eServer zSeries 890 , z9 EC, z9 BC, z10 EC, and z10 BC . It can also be used to provide key management on the IBM eServer zSeries 900 and IBM eServer zSeries 800.

AES operational key entry for the CEX2C on the z10 servers is available with TKE V5.3.

An operational key entry for the PCIXCC /CEX2C on the z990, z890, z9 EC, z9 BC, z10 EC, and z10 BC was introduced with TKE V4.1.

Crypto Express2 support is provided beginning with TKE 4.2 with FMID HCR7720. TKE 4.0 and 4.1 will always recognize a CEX2C as a PCIXCC.

The TKE workstation (Version 3 and Version 4) uses the IBM 4758 card. The TKE workstation Version 5 uses the IBM 4764 card.

Smart card reader

The smart card reader has a PIN pad, a display window, and two lights. On the PIN pad, the TKE smart card supports the numeric buttons (0-9), the red X cancel button, and the yellow <== backspace button.

Each smart card reader has two lights above the PIN pad: one green light and one yellow light. When you open a smart card application (SCUP, CNM or TKE), wait for the green light to come on for both smart card readers before you select a smart card function. The green light should always be on when the readers are attached.

Only one smart card application may be opened at a time. If more than one is opened, you will get an error message indicating that smart card functions are not available or that smart card readers are not available, depending on the application.

2.50 z10 multichip module (MCM)

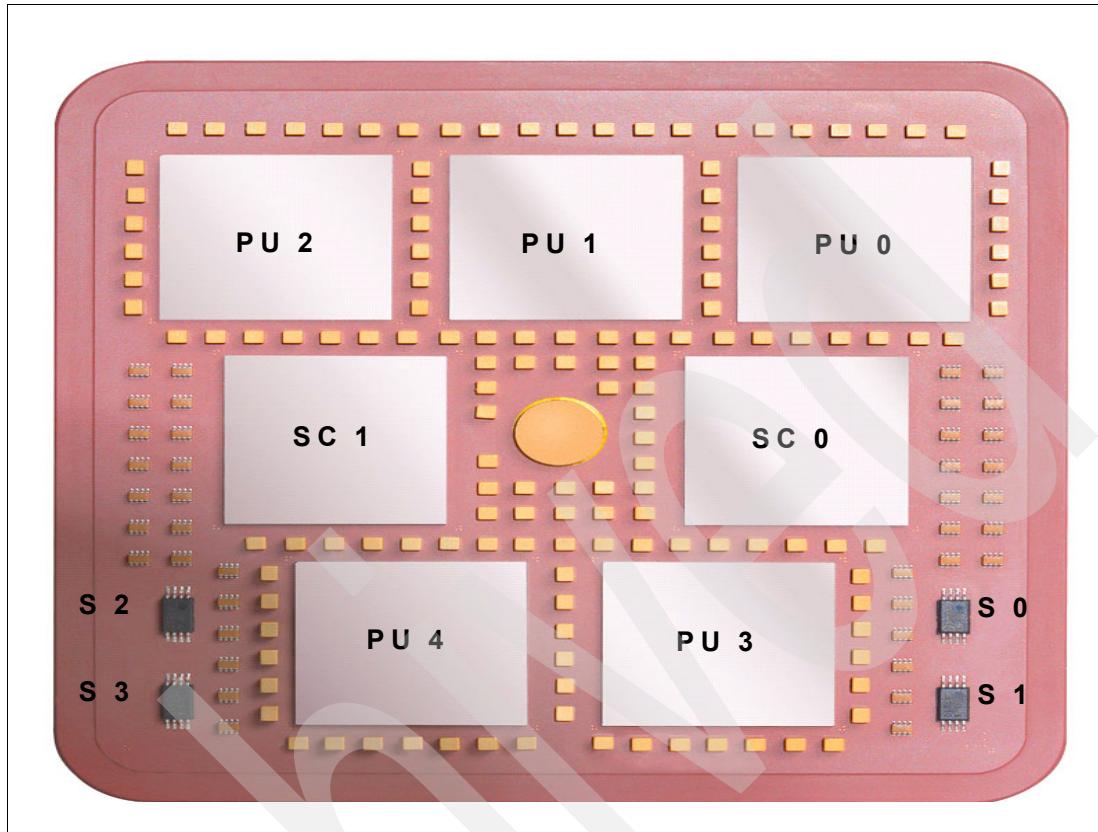


Figure 2-50 z10 multichip module

z10 multichip module

The z10 multichip module (MCM) provides a significant increase in system scalability and an additional opportunity for server consolidation. All books are interconnected with very high-speed internal communications links in a fully connected star topology through the L2 cache, which allows the system to be operated and controlled by the PR/SM facility as a memory-coherent symmetric multiprocessor (SMP).

MCM technology

The z10 EC is built on a proven superscalar microprocessor architecture. On each book there is one MCM. The MCM has five PU chips and two SC chips. The PU chip has up to four cores, which can be characterized as CPs, IFLs, ICFs, zIIPs, zAAPs, or SAPs. Two MCM sizes are offered, which are 17 or 20 cores.

Because of increased frequency (4.4 GHz versus 1.7 GHz), the chips on the MCM generate more heat than the z9 EC chips. The MCMs on the z10 EC therefore will continue to be modular refrigeration unit (MRU)-cooled with air backup.

Figure 2-50 on page 142 illustrates the CPACFs within the z10 hardware. Each z10 has a processor unit (PU) cage. It is located at the top of the A frame. The PU cage is populated by books. Each book has a certain amount of PUs and central storage.

Each MCM has five PU chips where the CPACF are located. Refer to 2.51, “z10 PU chip” on page 143 for more information about PU chips.

2.51 z10 PU chip

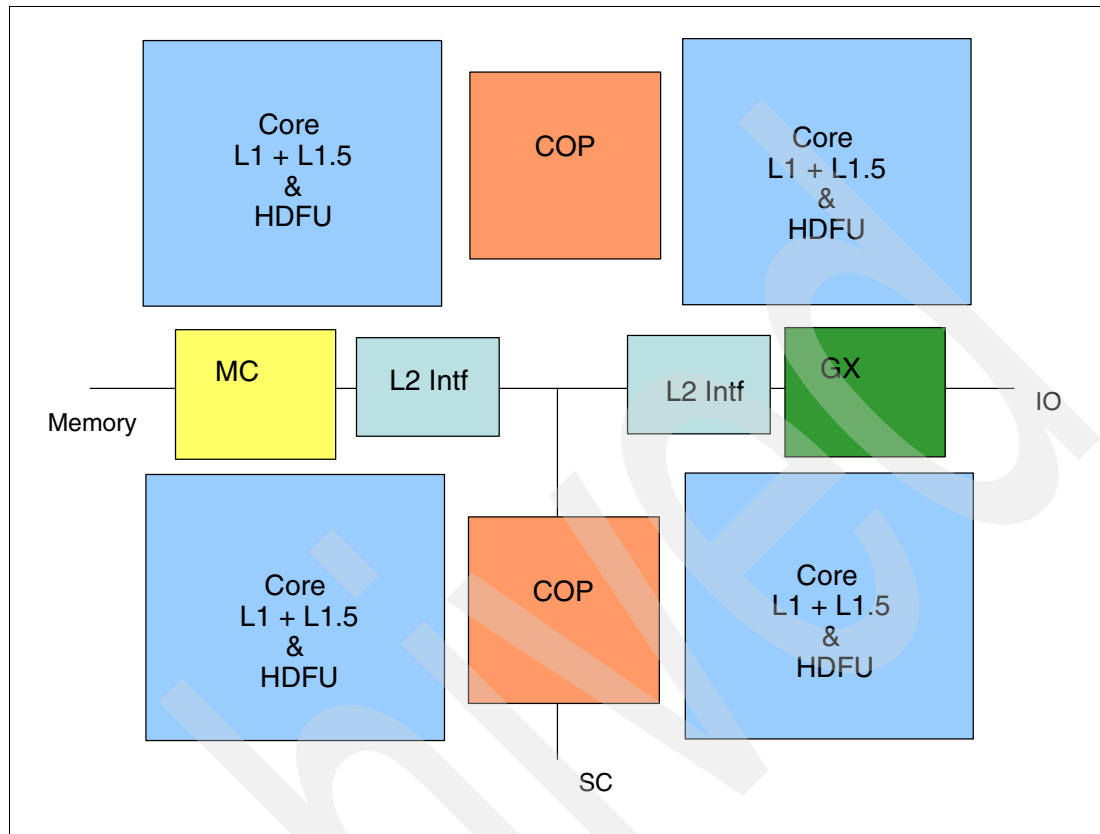


Figure 2-51 z10 PU chip

z10 PU chip

The z10 PU chip includes data compression and cryptographic functions such as the CP Assist for Cryptographic Function (CPACF). Hardware data compression can play a significant role in improving performance and saving costs over handling compression in software. Standard clear key cryptographic processors right on the processor translate to high-speed cryptography for protecting data in storage, integrated as part of the PU.

The z10 PU chip has two models:

- ▶ Four- PU model
- ▶ Three- PU model

Both models have two inbound co-processors containing the CPACF and the Ziv-Lempel data compression hardware. These functions are not located inbound of the PU for space and thermodynamic reasons. See 2.49, "Hardware crypto in a z10" on page 140 for more information about CPACF. The CPACF is shared between two PUs and dedicated to one PU in the three-PU model PU chip.

Because CPACF is a synchronous coprocessor, all its queue time and service time is include in the PU time invoking the CPACF. CPMF crypto counters have a significant amount of real-time information about CPACF performance. This will help you to verify bottlenecks in encrypting data.

2.52 CPAF crypto algorithms

- ❑ Data Encryption Standard (DES), which includes:
- ❑ Hashing algorithms such as SHA-1, and SHA-2
- ❑ Message authentication code (MAC)
- ❑ Pseudorandom number generation (PRNG)

Figure 2-52 CPAF crypto algorithms

CPACF crypto algorithms

Figure 2-52 on page 144 shows all the crypto algorithms executed by the CPACF inbound coprocessor. The z10 EC uses the Cryptographic Assist Architecture. The CP Assist for Cryptographic Function (CPACF) offers the full complement of the Advanced Encryption Standard (AES) algorithm and Secure Hash Algorithm (SHA). Support for CPACF is also available by using the Integrated Cryptographic Service Facility (ICSF). ICSF is a component of z/OS and can transparently use the available cryptographic functions CPACF, Crypto Express2, or Crypto Express3, to balance the workload and help address the bandwidth requirements of your applications.

Crypto algorithms

CPACF offers a set of symmetric cryptographic functions for high encryption and decryption performance of clear key operations for SSL, VPN, and data-storing applications that do not require FIPS 140-2 level 4 security. The cryptographic architecture includes support for:

- ▶ Data Encryption Standard (DES) data encryption and decrypting
- ▶ Triple Data Encryption Standard (triple DES) data encrypting and decrypting
- ▶ Advanced Encryption Standard (AES) for 128-bit, 192-bit, and 256-bit keys
- ▶ Pseudorandom number generator (PRNG)
- ▶ MAC message authorization
- ▶ Secure Hash Algorithm (SHA-1) hashing
- ▶ Secure Hash Algorithm (SHA-2) hashing (SHA-256, SHA-384, and SHA-512)

Data Encryption Standard (DES)

DES includes:

- ▶ Double-key DES (double DES)
- ▶ Triple-key DES (triple DES)
- ▶ Advanced Encryption Standard (AES) with secure encrypted 128-bit, 192-bit, and 256-bit keys (secure key AES is exclusive to System z10)

Hashing algorithms

The hashing algorithms that are included are:

- ▶ SHA-1
- ▶ SHA-2 support for : SHA-224, SHA-256, SHA-384, and SHA-512

Message authentication code (MAC)

In support of ISO 16609:2004, the cryptographic facilities support the requirements for message authentication, using symmetric techniques. The Crypto Express features provide the ISO 16609 CBC Mode T-DES MAC support. This support is accessible through ICSF callable services. ICSF callable services used to invoke the support are MAC Generate (CSNBMGN) and MAC Verify (CSNVMVR). The support includes:

- ▶ Single-key MAC
- ▶ Double-key MAC

Pseudorandom number generation (PRNG)

This support includes:

- ▶ Random number generation long (RNGL) with 8 bytes to 8096 bytes
- ▶ Random number generation (RNG) with up to 4096-bit key RSA support

2.53 CPMF crypto counters

- ❑ PRNG Function Count. This counter counts the total number of the PRNG functions issued by the CPU.
- ❑ SHA Blocked Cycle Count. This counter counts the total number of PU cycles blocked for the SHA functions issued by the PU because the SHA coprocessor is busy performing a function issued by another PU.
- ❑ DEA Cycle Count. This counter counts the total number of PU cycles when the DEA/AES coprocessor is busy performing the DEA functions issued by the PU.
- ❑ AES Blocked Function Count. This counter counts the total number of the AES functions that are issued by the PU and are blocked because the DEA/AES coprocessor is busy performing a function issued by another PU.

Figure 2-53 Some CPMF crypto counters

CPMF crypto counters

The crypto activity counter set collects information about activities caused by executing cryptographic operations in the CPACF. Because the z10 machine is the first in which two PUs share a cryptographic coprocessor, this counter set provides information about characteristics of the cryptographic workload and cryptographic interference between the two sharing PUs.

As we mentioned, CPACF consists of inbound cipher coprocessor for the DES and the AES and a hash coprocessor for the Secure Hash Algorithm. Both coprocessors can perform cryptographic operations simultaneously. However, when two sharing PUs attempt to use the same coprocessor, one will be blocked until a predetermined time slice has passed.

Additionally, a counter set is provided for each cryptographic coprocessor group to count cryptographic activities contributed by both of the sharing PUs. This counter set provides information about the utilization of each cryptographic coprocessor.

Crypto-Activity Counter Set

All counters in this counter set measures the specified activities of a CPACF contributed by this logical PU:

- ▶ PRNG Function Count. This counter counts the total number of the PRNG functions issued by the CPU.

- ▶ PRNG Cycle Count. This counter counts the total number of CPU cycles when the DEA/AES coprocessor is busy performing the PRNG functions issued by the CPU.
- ▶ PRNG Blocked Function Count. This counter counts the total number of the PRNG functions that are issued by the CPU and are blocked because the DEA/AES coprocessor is busy performing a function issued by another CPU.
- ▶ PRNG Blocked Cycle Count. This counter counts the total number of CPU cycles blocked for the PRNG functions issued by the CPU because the DEA/AES coprocessor is busy performing a function issued by another CPU.
- ▶ SHA Cycle Count. This counter counts the total number of PU cycles when the SHA coprocessor is busy performing the SHA functions issued by the PU.
- ▶ SHA Blocked Function Count. This counter counts the total number of the SHA functions that are issued by the PU and are blocked because the SHA coprocessor is busy performing a function issued by another PU.
- ▶ SHA Function Count. This counter counts the total number of the SHA functions issued by the CPU.
- ▶ SHA Blocked Cycle Count. This counter counts the total number of PU cycles blocked for the SHA functions issued by the PU because the SHA coprocessor is busy performing a function issued by another PU.
- ▶ DEA Function Count. This counter counts the total number of the DEA functions issued by the PU.
- ▶ DEA Cycle Count: This counter counts the total number of PU cycles when the DEA/AES coprocessor is busy performing the DEA functions issued by the PU
- ▶ DEA Blocked Function Count. This counter counts the total number of the DEA functions that are issued by the PU and are blocked because the DEA/AES coprocessor is busy performing a function issued by another PU.
- ▶ DEA Blocked Cycle Count. This counter counts the total number of PU cycles blocked for the DEA functions issued by the PU because the DEA/AES coprocessor is busy performing a function issued by another PU.
- ▶ AES Function Count. This counter counts the total number of the AES functions issued by the PU.
- ▶ AES Cycle Count. This counter counts the total number of PU cycles when the DEA/AES coprocessor is busy performing the AES functions issued by the PU.
- ▶ AES Blocked Function Count. This counter counts the total number of the AES functions that are issued by the PU and are blocked because the DEA/AES coprocessor is busy performing a function issued by another PU.
- ▶ AES Blocked Cycle Count. This counter counts the total number of PU cycles blocked for the AES functions issued by the PU because the DEA/AES coprocessor is busy performing a function issued by another PU.

2.54 Crypto counters example

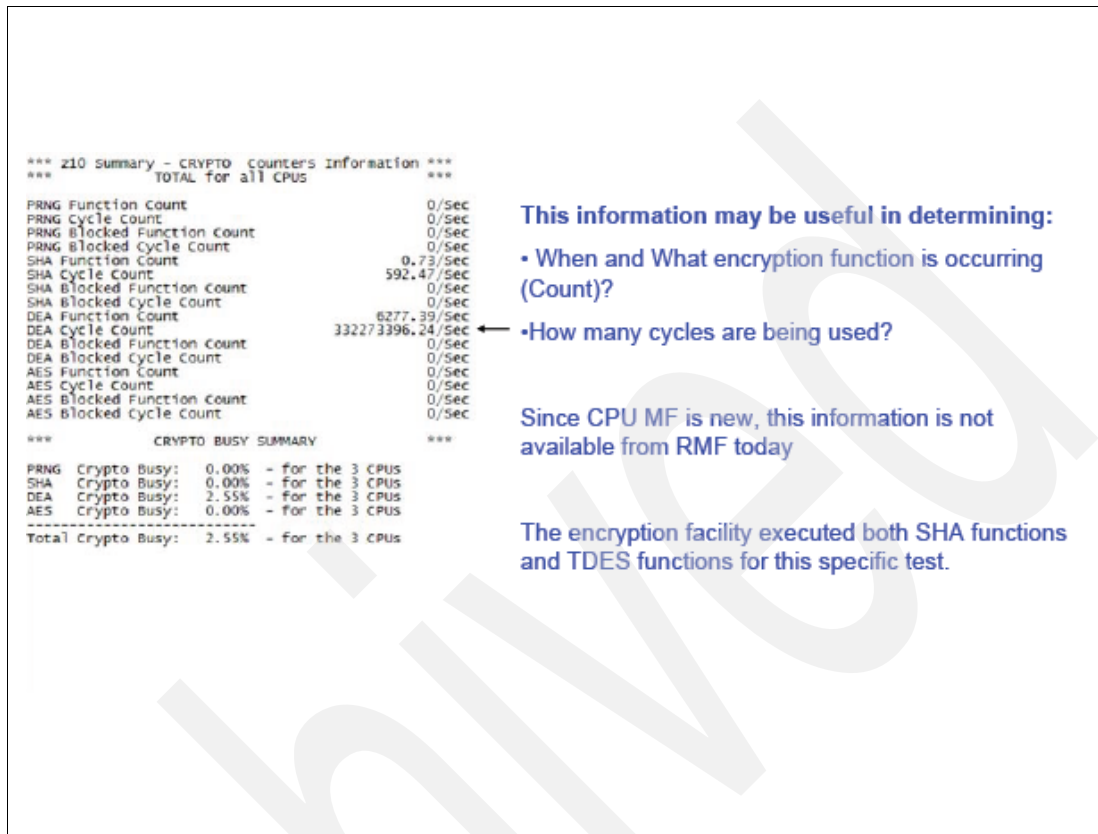


Figure 2-54 Crypto counters example

Crypto counters example

Figure 2-54 on page 148 shows a practical example of running HIS together with CPMF producing crypto counters values.

- ▶ **DEA Function Count.** This counter counts the total number of the DEA functions issued by the PU. The value shown here is 6777.39.
- ▶ **DEA Cycle Count.** This counter counts the total number of PU cycles when the DEA/AES coprocessor is busy performing the DEA functions issued by the PU. The value shown here is 332273396.24.

If you multiply this figure by 0.23 nanosecods (z10 EC cycle time), you get 7.6 PU milliseconds along the observation interval.

If you divide this number by the DEA Function Count, you get the amount of PU per DEA request, namely 1.1 microseconds.

- ▶ The derivation of the total crypto utilization is equal to 2.55%.

2.55 CPU-measurement sampling facility

- ❑ The CPU-measurement sampling facility is a CPMF component. It provides a means to take a snapshot of the logical PU at a program-specified sampling interval. This function is installation controlled by the Parmlib keywords:
 - **SAMPFREQ.** This sets the sample frequency. The default value is 800,000 samples per minute, or 13,333 per second.
 - **DURATION.** This indicates for how long the sampling is active. The default is 10 minutes.

Figure 2-55 CPU-measurement sampling facility

CPU-measurement sampling facility

The CPU-measurement sampling facility is a CPMF component. It provides a means to take a snapshot of the logical PU at a program-specified sampling interval. Each snapshot produces a set of sample data, which includes the instruction address of an instruction being executed and some state information about the PU. To activate this function the z/OS HIS component should issue the Set Program Parameter (B280) instruction.

At the end of each sampling interval, the PU stores sample data into measurement blocks allocated by z/OS. On top of that, z/OS can set up alerts (triggered by external interrupts) before the measurement blocks are filled up so that it can take appropriate actions to save the sampling data and to free up space in these measurement blocks.

The CPU-measurement sampling facility allows tooling programs to map instruction addresses into modules or tasks, and it facilitates the determination of hotspots. It is a tracing mechanism, and to execute this function a load module memory map must be built. This function is activated by setting the sampling authorization control in the HMC.

This function is installation-controlled by the MODIFY HIS:

- ▶ **SAMPFREQ.** This sets the sample frequency. The default value is 800,000 samples per minute, or 13,333 per second. A very small sampling interval would severely impact system performance and significantly distort sample data. A reasonable sampling interval should be used.
- ▶ **DURATION.** This indicates for how long the sampling is active. The default is 10 minutes.

2.56 CPU-measurement sampling facility functions

- ❑ The facility includes two sampling functions:
 - Basic sampling. This is for ordinary use as generating data to produce better performance.
 - Diagnostic sampling. This is intended for problem determination.
- ❑ The sampling function can be in one of the following states:
 - Unauthorized
 - Disabled
 - Inactive
 - Active

Figure 2-56 CPU-measurement sampling facility functions

CPU-measurement sampling facility functions

The CPU-measurement sampling facility includes two sampling functions:

- ▶ Basic sampling
- ▶ Diagnostic sampling

Both functions use the same sampling control registers, the same sampling buffer structure, the same external interruption events, and the same instructions. The main difference between these two functions is the nature of the sample data.

- ▶ The basic sampling function is for ordinary use as generating data to be used to produce a better program performance,
- ▶ The diagnostic sampling function is intended for problem determination; it can be authorized by IBM Service Personnel. The sample data provided by the basic sampling function is not included in the sample data provided by the diagnostic sampling function. To obtain meaningful diagnostic sampling data, both sampling functions will be activated.

A more detailed discussion of diagnostic sampling data is beyond the scope of this publication.

Note that the sample data size and format for each sampling function are model-dependent.

Sampling function possible states

The sampling function can be in one of the following states:

- Unauthorized** When a sampling function is in the unauthorized state, the function cannot be used and no sample data is stored. An external means is provided to authorize or unauthorize the use of the function.
- Disabled** When a sampling function is in the disabled state, the function is authorized for use but the program has not enabled the function yet. In this state: no new sample data is stored; the contents of the sample-data blocks remain unchanged; and no sampling control, except for authorization controls, is preserved.
- Inactive** When a sampling function is in the inactive state, the function is authorized, enabled, and deactivated. In this state, no new sample data is stored, the contents of sample data blocks remain unchanged, and sampling controls are preserved and can be extracted.
- Active** When a sampling function is in the active state, the function is authorized, enabled, and activated. In this state, new sampling data is stored during each sampling interval and sampling controls can be extracted.

When the logical CPU enters the stopped state from the operating state (an example can be a LPAR intercept), active sampling functions are stopped. When the CPU enters the operating state from the stopped state (as an LPAR SIE instruction), sampling functions resume the states they were in when they were last stopped.

2.57 CPMF sample data block tables

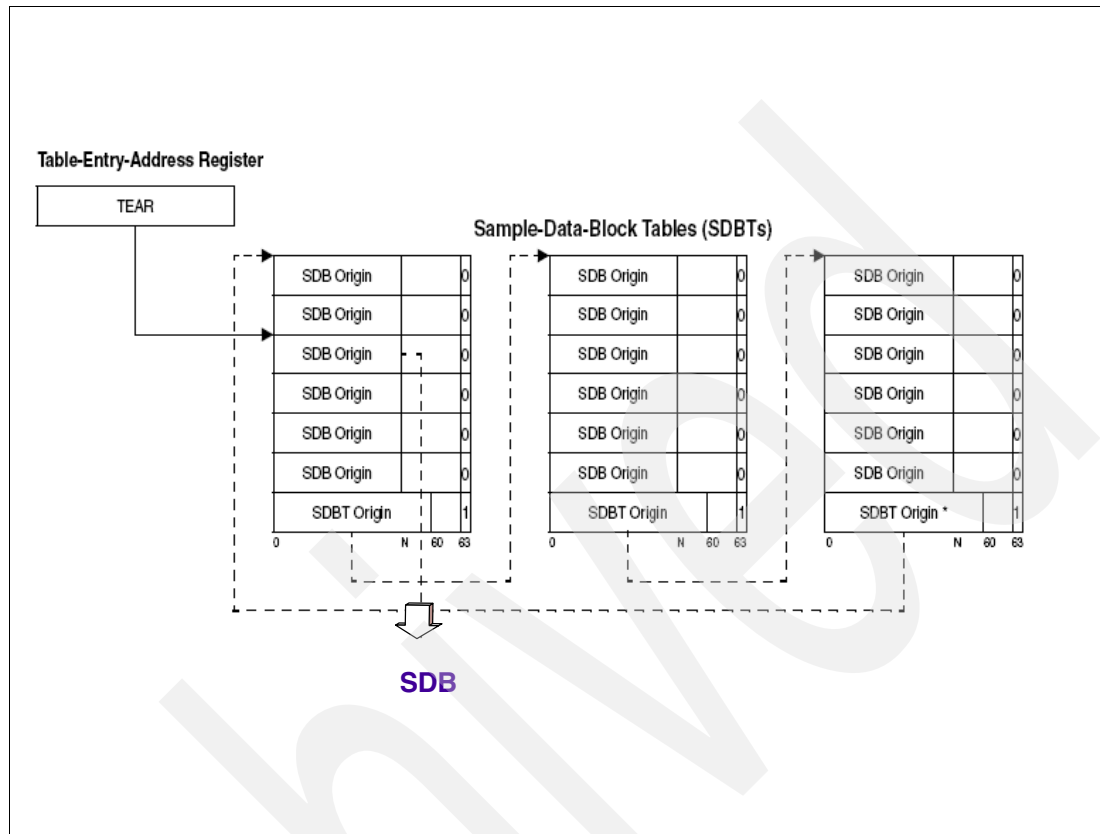


Figure 2-57 CPMF sample data block tables

CPMF sample data block tables

The CPU measurement sampling facility stores sample data in sample data blocks (SDBs).

z/OS allocates a number of SDBs in central storage. These blocks are used by the CPU-measurement sampling facility to store sample data during each sampling interval. See 2.58, “CPMF sample data block (SDB)” on page 153 for more information about SDBs.

SDBs are pointed to by entries in sample data tables. Each SDB is designated by a 8-byte block link entry in a sample data block table (SDBT), as shown in Figure 2-57 on page 152.

The sample data block tables are chained together through SDBT origin pointers.

The current entry of the sample data block table is designated by the contents of the TEAR, or table-entry-address register.

2.58 CPMF sample data block (SDB)

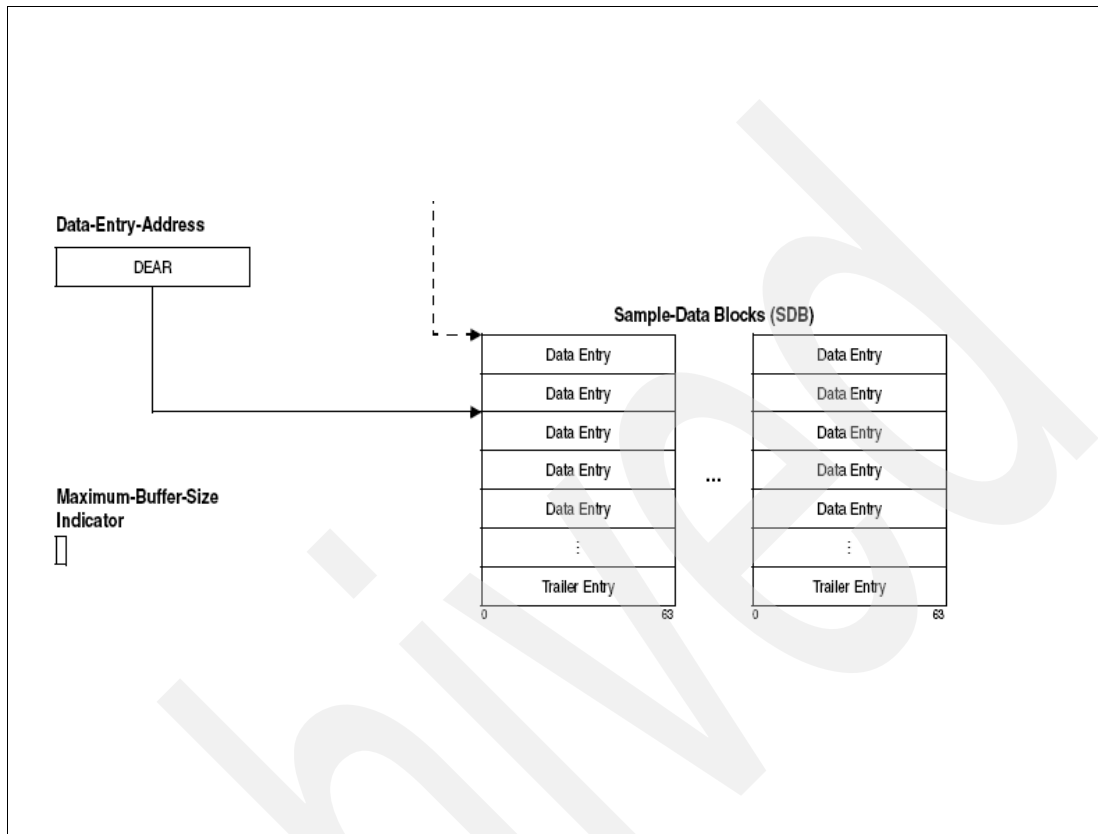


Figure 2-58 Sample data blocks (SDBs)

Sample data blocks

Sample data blocks (SDBs) are used by the CPMF CPU-measurement sampling facility to store sample data during each sampling interval. A set of SDBs are defined by the operating system in central storage. SDBs are pointed by entries in SDB tables.

A SDB is composed of data entries where the sample data is really stored.

The next data entry of the SDB is designated by the contents of the DEAR, or data entry address register. See 2.59, "Sample data block data entry" on page 154 to view the contents of such a data entry.

2.59 Sample data block data entry

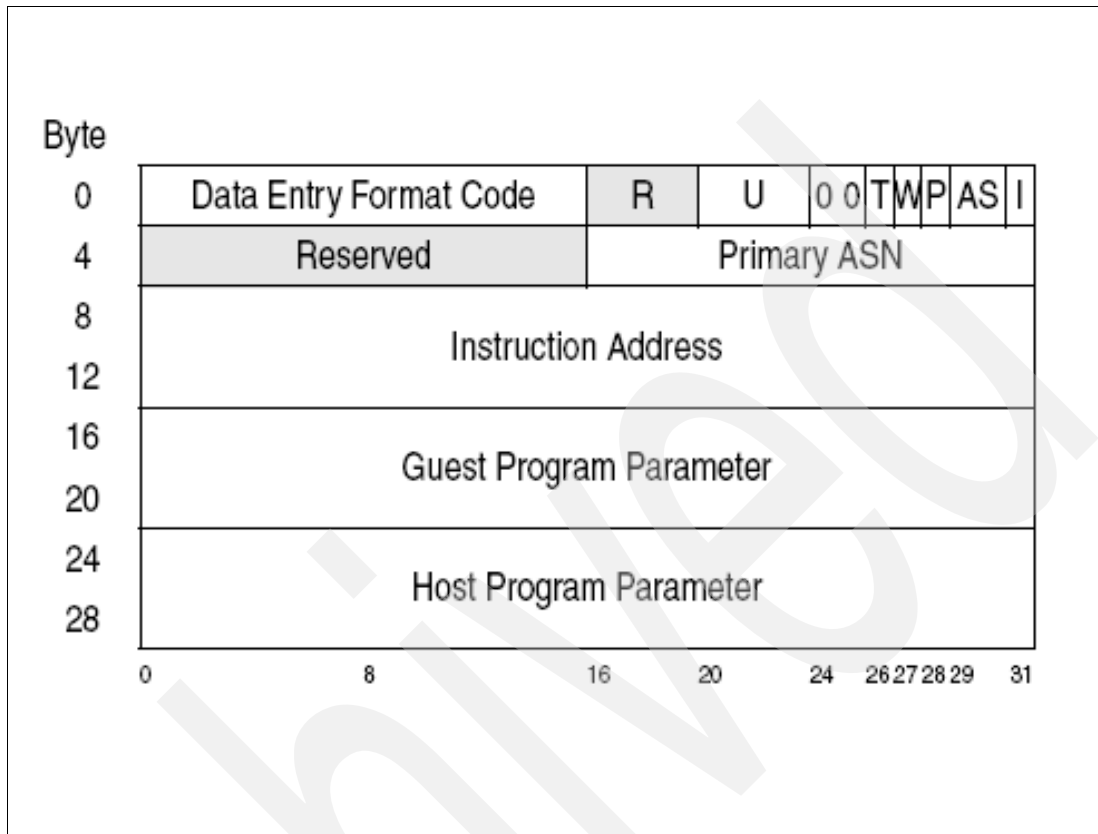


Figure 2-59 Sample data blocks data entry layout

Sample data blocks data entry

Figure 2-59 on page 154 illustrates the contents of the data entry.

- ▶ Data Entry Format Code. This is the format code of the data entry.
- ▶ Reserved (R). This is reserved for programming use. Zeroes are stored in this field by the sample data block update process.
- ▶ Number of Unique Instructions (U). These specify the number of unique, completed instructions that were executed simultaneously during the sampling cycle when the unique cycle indicator is on. A pipelined PU (as z10) can execute multiple instructions concurrently in an overlapped fashion. Furthermore, on some models, each stage of a pipelined PU can execute multiple instructions simultaneously. During an instruction execution, a unique cycle indicator is turned on for one cycle at the sampling point, which is the place in the PU that the sample data is taken from. The sampling point depends on the model but is the same for all instructions executed on the same model. For a pipelined PU, the sampling point is usually a particular pipeline stage. Depending on the model, it is unpredictable when the unique cycle indicator is turned on during an instruction execution.

The contents of this field can be used to estimate cycles per instruction when a sufficiently small sampling interval and an adequately larger number of samples are used. The cycles per instruction for a particular measurement can be estimated by dividing the number of busy samples (samples with the wait state bit set to zero) by the total number of unique instructions in all busy samples. Remember that CPAI can also be calculated through the CPMF CPU-measurement counter facility as described in 2.29, “How CPMF works” on page 106.

- ▶ **DAT Mode (T).** This is the DAT mode bit in the PSW of the PU telling if virtual address translation is being executed.
- ▶ **Wait State (W).** The wait state bit in the PSW of the PU is on, meaning that z/OS entered a voluntary wait because there is no TCB or SRB to be executed at a z/OS logical partition.
- ▶ **Problem State (P).** The problem state bit in the PSW of the PU is on, meaning that the executing program does not belong to the z/OS kernel. That is, the PU is not able to execute privileged instructions.
- ▶ **Address-Space Control (AS).** This is the address space control in the PSW of the PU. It tells the addressing mode, which can be primary, secondary, access register, or home addressing mode.
- ▶ **Invalid Indication (I).** This indicates whether the entry is valid or invalid. When the bit is zero, the entry is valid. When the bit is one, the entry is invalid. An entry is set to invalid when sample data in the entry are inconsistent.
- ▶ **Primary ASN.** This is the primary ASN in bits 48-63 of control register 4 of the PU.
- ▶ **Instruction Address.** This is the instruction address of an instruction that the PU was executing during the sampling cycle. Instruction addresses are treated as:
 - Real addresses in the real mode (DAT bit off).
 - Primary virtual addresses in the primary-address mode, secondary-space mode, or access-register mode, and as home virtual addresses in the home-space mode.

When at the sampling point, the PU is executing multiple instructions simultaneously during the sampling cycle, but only the address of one instruction among these simultaneously executed instructions is reported. The selection of which instruction address to be reported is model-dependent. When the wait state bit is one, the contents of this field are unpredictable. When a sampling occurs and if the sampling point is not executing any instruction because of delay in some other pipeline stage, it is unpredictable which address of the instructions being executed concurrently in the CPU is reported.

Guest parameters

There are two guest parameters:

- ▶ **Guest Program Parameter.** The guest program parameter can be used to identify the individual task that contributed to the sample data. If the CPU is running at the virtual machine level (z/VM) when sampling occurs, then:
 - If the host indicator is one at the logical partition level, byte offsets 16-23 of the data entry contain the program parameter most recently set by the PU at the virtual machine level.
 - If the host indicator is zero, the contents of byte offsets 16-23 are unpredictable.
 If the PU is running at the logical partition (no z/VM) level when sampling occurs, then:
 - If the host indicator is set to zero, byte offsets 16-23 of the data entry contain the program parameter most recently set by the CPU at the logical partition level.
 - If the host indicator is one, the contents of byte offsets 16-23 are set to zeros.
- ▶ **Host Program Parameter.** When sampling occurs, if the host indicator is:
 - One at the logical partition level, then byte offsets 24-31 of the data entry contain the program parameter most recently set by the CPU at the logical partition level.
 - Zero, then the contents of byte offsets 24-31 are set to zeros.

2.60 I/O performance and metrics SLA

```

      S H A R E D   D I R E C T   A C C E S S   D E V I C E   A C T I V I T Y
z/OS V1R5          SYSPLX RMFLP4X          DATE 07/28/2004          INTERVAL 15.56.594
                   RPT VERSION V1R5 RMF          TIME 16.15.14          CYCLE 01.000 SECONDS
TOTAL SAMPLES (AVG) = 900.0 (MAX) = 900.0 (MIN) = 900.0

```

DEV	DEVICE	VOLUME	PAV	SMF	SYS	IODF	LCU	ACTIVITY	AVG	AVG	AVG	AVG	AVG	AVG	AVG	%	%	%	AVG
NUM	TYPE	SERIAL		ID	SUFF			RATE	RESP	IOSQ	CMR	DB	PEND	DISC	CONN	DEV	DEV	DEV	NUMBER
									TIME	TIME	DLY	DLY	TIME	TIME	TIME	CONN	UTIL	RESV	ALLOC
8004	33903	GG8004		*ALL				4043.964	1.0	0.0	0.2	0.0	0.4	0.0	0.6	33.03	33.03	0.0	8.9
8	SYS1	29	00A9	1139.045					1.0	0.0	0.2	0.0	0.4	0.0	0.7	9.33	9.33	0.0	3.0
8	SYS2	29	00A9	1466.329					1.0	0.0	0.2	0.0	0.4	0.0	0.7	11.96	11.96	0.0	3.0
8	SYS3	29	00A9	1438.591					1.0	0.0	0.2	0.0	0.4	0.0	0.7	11.74	11.74	0.0	2.9
7640	33903	GG7640		*ALL				10.869	3.0	0.1	10.0	0.7	0.9	0.2	1.8	1.97	2.16	0.2	125
1	SYSF	29	00A6	1.098					6.2	0.0	0.0	3.1	3.4	0.4	2.4	0.26	0.31	0.0	36.0
1	SYS1	29	00A6	9.771					2.6	0.1	0.0	0.4	0.6	0.1	1.8	1.71	1.85	0.2	88.5
7641	33903	GG7641		*ALL				0.607	1.9	0.0	0.0	0.2	0.8	0.0	1.1	0.07	0.07	0.0	15.0
1	SYSF	29	00A6	0.201					2.8	0.0	0.0	0.5	1.8	0.0	1.0	0.02	0.02	0.0	1.0
1	SYS1	29	00A6	0.406					1.5	0.0	0.0	0.0	0.3	0.0	1.1	0.05	0.05	0.0	14.0

Figure 2-60 RMF shared DASD activity report - extract

I/O performance and metrics

In this section we discuss performance management considerations and metrics associated specifically with an I/O operation.

As CPU speed increases, the I/O response time (I/O Tr) becomes the determinant factor in the average transaction response time. I/O response time is shown in the following formula:

$$I/O \text{ Tr} = I/O \text{ Ts} + I/O \text{ Tw}$$

Excellent transaction response time gains can be achieved by reducing I/O wait time (Tw) and I/O service time (Ts). Turning into I/O performance metrics, the I/O Tr is formed by the following components:

$$I/O \text{ Tr} = \text{IOSQ Time} + \text{Pending Time} + \text{Connect Time} + \text{Disconnect Time}$$

Where:

$$I/O \text{ Tw} = \text{IOSQ Time} + \text{Pending Time}$$

$$I/O \text{ Ts} = \text{Connect Time} + \text{Disconnect Time}$$

The following list explains these terms in greater detail.

I/O Queue Time	This refers to the time waiting for device availability in the z/OS Operating System.
Pending time	This refers to the time from the issuance of the SSCH instruction until the start of the dialog between the channel and I/O controller. It is caused by having I/O path (as channel, controllers) elements busy.
Disconnect time	This refers to the period of time in which the I/O operation is already started but the channel and I/O controller are not in a dialog.
Connect time	This refers to the period of time in which the channel is transferring data from or to the controller cache or exchanging control information with the controller about one I/O operation.

Figure 2-60 shows the RMF shared DASD activity report. The times for volser GG8004 are displayed in line *ALL. Numerically, it is shown as follows:

$$1.0 = 0.0 + 0.4 + 0.6 + 0.0 \text{ (roughly)}$$

Note that there are other metrics, such as the following:

- ▶ I/O Intensity = I/O Tr * I/O Rate
- ▶ I/O Traffic = I/O Ts * I/O Rate
- ▶ I/O Density = I/O Rate per DASD space capacity

Techniques to reduce the I/O Tr

Refer to *VSAM Demystified*, SG24-6105, for information about how to decrease I/O tr times. Generally speaking, you can reduce the average I/O response time (I/O Tr) using software or hardware techniques.

Software techniques

Software techniques aim to decrease average I/O response time by reducing the number of I/O operations, or bytes transferred, by implementing *buffering*. Buffering decreases the number of bytes transferred in a random access and make I/Os more efficient for sequential access.

Buffering can be implemented through:

- Virtual address spaces (above and below the bar)
- Data spaces
- Hiperspaces

There are other software techniques to reduce the I/O response time, such as:

- Data compression, which decreases the number of bytes transferred in any type of access
- Data striping, which makes I/Os more efficient for sequential accesses, due to parallelism

Hardware techniques

The following hardware techniques allow you to reduce I/O response time:

- ▶ Faster channels (such as FICON Express)
- ▶ Faster device paths (adapters) to the controllers

- ▶ Using a larger and more efficient cache controller
- ▶ More DASD subsystem concurrency, for example, parallel access volumes (PAV) in DS8000
- ▶ Faster disks (RPMs), small size, RAID-10 and solid state device

Experience has shown that when you overcome an I/O bottleneck and increase the I/O rate (for example, by implementing data striping), then the bottleneck moves from the I/O subsystem to the processor, which is, however, at another performance level.

2.61 z/OS CPU time accounting

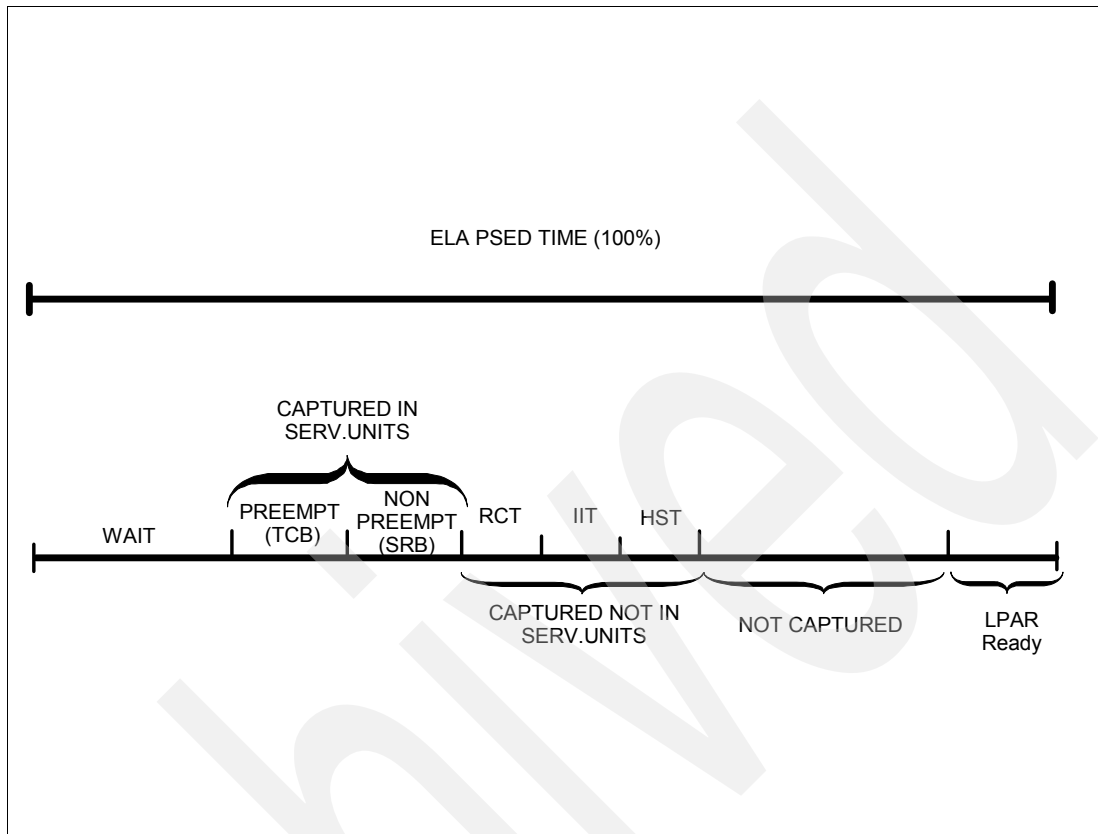


Figure 2-61 z/OS CPU time accounting

z/OS CPU time accounting

Transaction CPU time is extensively explained in other parts of this publication. This section describes how z/OS captures such information.

An important aspect of performance management is capturing performance data for report generation. z/OS collects CPU time consumed by the tasks (TCBs) and service requests (SRBs) associated to executed transactions. When a TCB or an SRB is dispatched, an X value is loaded in the CPU timer (a hardware clock that decreases its contents). When the TCB or SRB is interrupted, the current CPU timer (X - Y) is stored in memory.

Y is then accumulated in the ASCB control block (or in an enclave, ENCB), and therefore in an AS or enclave basis. There are two counters in ASCB:

- ▶ One for preemptible processes, such as TCB, SRB client, SRB enclaves
- ▶ One for non-preemptible processes, such as SRB traditional

See 2.62, “z/OS preemptability” on page 161 for more detailed information about preemptible and non-preemptible processes. Note that all information referring here to CPUs also apply to zIIPs and zAAPs.

The accumulated time (in an AS or enclave basis) is called *captured* time and is converted into CPU service units. See 2.19, “CPU service units” on page 85 for more information about this topic.

Wait% time

It also accounts for the CP wait time when the logical CPU is placed in wait state because there is no dispatchable unit (TCB or SRB) to be executed. In that case, *wait* time is the time that the logical CP is in wait state due to lack of activity in the logical partition.

LPAR Ready%

There is also the possibility for the installation to derive LPAR Ready%. This can be done by subtracting MVS Busy% minus LPAR Busy% in the RMF CPU report. LPAR Ready% is the percentage of time that you have ready logical PUs in the logical partition (LP). Or, the LPAR does not totally serve the demand in this LP. This occurs when logical PUs in the LP are ready to execute but because of demand in other LPs, the LPAR is not delivering physical PUs to the LP's logical PUs. It appears that the logical PU is stopped. LPAR Ready% is the measurement of LPAR contention in the LP. If this difference is considerable and this LP is important to your business, one solution is to raise the weight or the number of logical PUs, or both.

There is also a captured time that is *not* converted in CPU SUs. It is the captured CPU time not included in ASCB preemptible or non-preemptible counters, and consequently not converted to CPU SUs. There are three elements generating such times:

- ▶ IIT, the CPU time spent in an I/O SLIH routine
- ▶ RCT, the CPU time spent in a Region Control task
- ▶ HST, the CPU time spent accessing hiperspaces

Refer to Figure 2-15 to see those numbers in the RMF - Workload Activity Report. Keep in mind, though, that the RMF report is a *global* report but those times are captured in a *splex* scope. The value 9.5% in field % APPL is derived through the formula:

$$\% \text{ APPL} = \text{TCB} + \text{SRB} + \text{RCT} + \text{IIT} + \text{HST} / \text{RMF Interval} * 100$$

% APPL indicates the percent of CPU consumed. Therefore, a number such as 150% can be interpreted as saying that the workload consumed 1.5 CPUs.

Non-captured CPU time

There are also other activities executed in the CPU that are not accounted by z/OS. This time is called *non-captured* time. It can be numerically derived by subtracting the captured time from the total busy time. Non-captured time includes:

- ▶ Overhead, such as page faults and swaps
- ▶ Global services, such as Dispatcher, WLM sampling, and SRM, where there is no address space or enclave in charge
- ▶ Services, where it is impossible to find the requested task (such as I/O FLIH)
- ▶ LPAR processing associated with this logical partition

Capture ratio

Non-captured time needs to be apportioned among the address spaces and enclaves to be calculated in total usage. Therefore, the concept of *capture ratio* was introduced. Capture ratio is the result derived from the formula:

$$\text{Capture ratio} = \text{Captured_time} / \text{Total_busy_time}$$

Capture ratio is used in capacity planning and accounting to distribute the non-captured time proportionally to address spaces and enclaves.

Dividing the address space captured time by the capture ratio gives an approximation for the time indirectly consumed by these address spaces or enclaves, including the non-captured time.

2.62 z/OS preemptability

- ❑ Reasons for being non-preemptible:
 - Having traditional SRB
 - Reduced preemption that avoids too much save/restore dispatchable units state
 - Program in the dispatchable unit running disabled for interrupts

Figure 2-62 Reasons for being non-preemptable

z/OS preemptability

Preemptible processes are the dispatchable units capable of being suspended (placed in ready state) if another dispatchable unit with a higher dispatching priority becomes ready for executing in the CPU.

In most cases, z/OS is very responsive to the fact that a high dispatching priority TCB/SRB became ready (simply being delayed by CPU), preempting a low dispatching priority TCB/SRB.

Being a preemptible operating system has the advantage of honoring first the high priority transactions (the ones important to business). This is key, mainly when the CPU utilization is very high.

A preemptible operating system has the disadvantage of introducing overhead in the system because preempting a TCB/SRB implies saving the status of the preempted TCB/SRB, and restoring later its status.

Putting all this together, there are a few cases where z/OS is not preemptible, such as:

- ▶ Traditional SRBs are always non-preemptible. They are designed to execute a small piece of code and then finish. In this case, it is desirable to let the SRB proceed from start to finish without incurring the overhead of saving/restoring status. However, there is some “dishonest code” that runs in traditional SRB mode, which are large pieces of programs enjoying the lack of preemptibility.

Today there are other types of preemptible SRBs, such as enclave SRBs and client SRBs.

- ▶ Reduced preemption. When a task gains control of the CPU, z/OS guarantees the use of the processor for a certain minimum amount of time. This time justifies the z/OS time investment to restore its status. It means that during this time, this task is non-preemptible. There is an exception is if the requesting high priority task has a dispatching priority of 255.
- ▶ When the CPU is running disabled for interrupts due to integrity reasons. This is a rare occurrence.

2.63 Formulas and laws in performance management

- Little's Law
- Markov's Equation
- Pareto's 80/20 Rule
- Partition's Law
- Law of Diminished Returns
- Principle of Locality

Figure 2-63 Formulas and laws in performance management

Formulas and laws in performance management

This section describes laws, rules, and principles that are frequently used in performance management. The criteria for selection is the importance towards performance analysis and simplicity; you do not need to have a mathematical degree to use them in a real-life performance analysis task.

Next, we look at these formulas and laws in more detail starting with 2.64, “Little’s Law” on page 164.

2.64 Little's Law

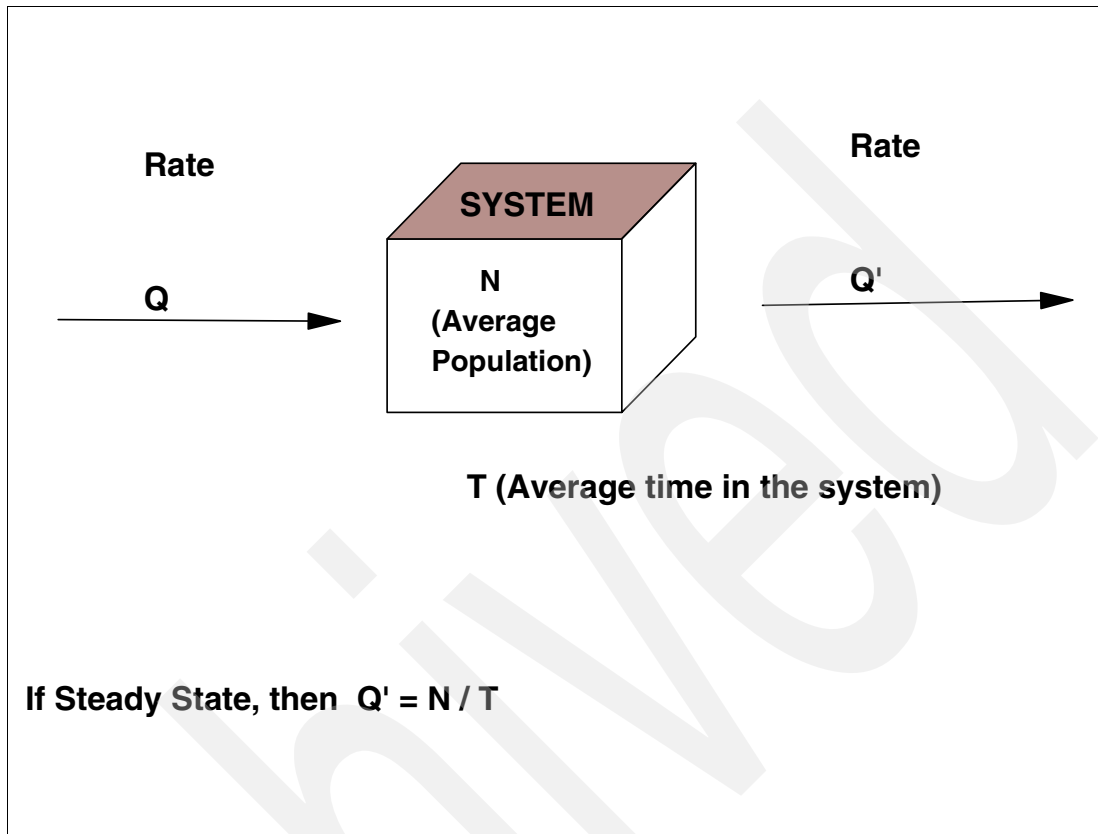


Figure 2-64 Little's Law

Little's Law

In Little's Law, a system (such as a computer or a bank branch office) enters elements (such as transactions in a computer or customers in a bank branch office) at a specific rate Q and leaving at a specific rate Q' . T is the average time spent by each element in the system, and N is the average population of elements in the system. If the system is in steady state, then, Q' is equal to N/T . In an approximation, we may say that in this state $Q=Q'$.

In other words, the average number of things in the system is the product of the average rate at which things leave the system and the average time each one spends in the system, and if there is a gross "flow balance" of things entering and leaving, the exit rate is also the entry rate. Peter Denning succinctly phrases this rule as "The average number of objects in a queue is the product of the entry rate and the average holding time."

Little's Law applications

There are many applications for Little's Law, such as the following formula:

$$Q = N / (T_r + T_t)$$

Where:

- N** This is the average number of users sending transactions (logged on).
- T_r** This is the response time.
- T_t** This is the average "thinking time" of these users.

The following formula is used by RMF to capture the average IOS queue time along an I/O operation. This IOS queue time is the time that the I/O request spends in average in the software queue being delayed because a previous I/O request towards the same device, coming from this system, did not finish yet:

$$\text{Avg_IOSQ TIME} = \text{IOS_Avg_Queue_Length} / \text{IOs_Rate}$$

Archived

2.65 Markov's Equation

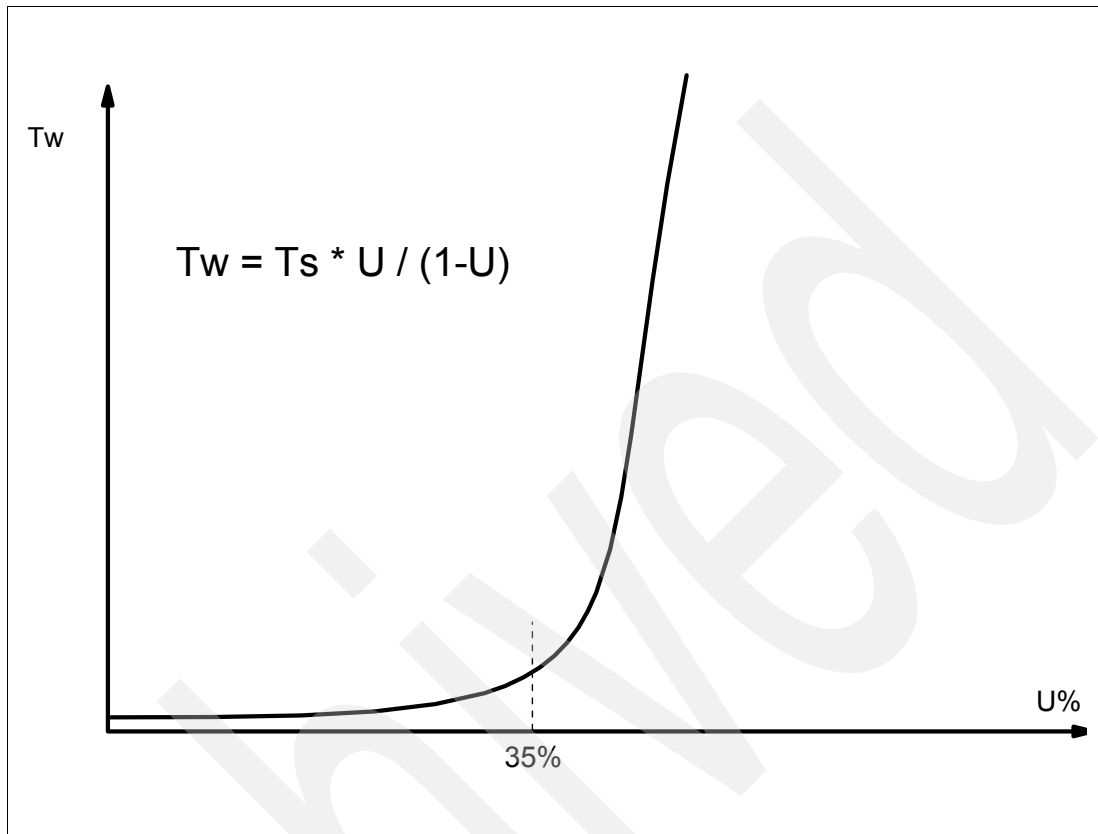


Figure 2-65 Markov's Equation curve

Markov's Equation

Markov's Equation shows the function $T_w = f(T_s, U)$ that is, the relation of Average Wait Time (T_w) with Service Time (T_s) and the Utilization (U):

$$T_w = T_s * U / (1 - U)$$

The conditions where the formula applies are:

- ▶ There is only one server.
- ▶ The system is in steady state.
- ▶ The distribution of the interarrival time (the time between two consecutive transactions using the server) is exponential.

This is a simplification of the queuing theory described in the M/M/1 formula, where both the interarrival time and the service time are exponential. (For additional details on the M/M/1 formula, refer to IBM Redpaper publication *A Solution to the Multiserver Priority Queuing Model*, REDP-3969).

Figure 2-65 on page 166 is a graphical depiction of Markov's equation. As you can see, for utilizations greater than 35% the T_w raises drastically, thereby causing a degradation in T_w and consequently in T_r (average response time).

When U is trending to one, T_w trends to infinity. This means that if an operating system is unable to block transactions from entering the system, when the demand is unaffordable, the

system may soon run out of resources. When CPU reaches 100%, memory could be exhausted by so many ongoing transactions.

z/Os and Markov

z/OS, unlike other operating systems, has such mechanisms of workload rejection. Many z/OS installations have been working close to an average CPU utilization of 100% for a long time, without needing an IPL although sometimes you might assist with a failure, mainly due to serialization.

In regard to CPU resource, when the average utilization of greater than 35% in z/OS and the critical work (z/OS itself and online applications, for example) consumes up to 35%, all the increase in T_w is experienced by the rest of the low priority transactions (such as batch). This is mainly true because z/OS tries to respect the values of the dispatching priorities. In other words, it tries to be a preemptible operating system. In this context, *preemptibility* means respecting the dispatching priorities of the dispatchable units (TCB/SRB) associated with the transactions.

Referring to Figure 2-65, if the service time is constant, you can observe that when the $U\%$ is greater than the 35% (knee point) on your curve, then wait time increases drastically. As a consequence, in a z/OS system, if the key business transactions (highest dispatching priorities) and the z/OS tasks use more than the knee point on the curve, then the T_w for these transactions starts to increase, causing performance degradation (high T_r) in those critical workloads. If that is not the case, then the rest of the capacity can be consumed by other workloads, up to 100%, without impacting the business.

2.66 DASD saturation point

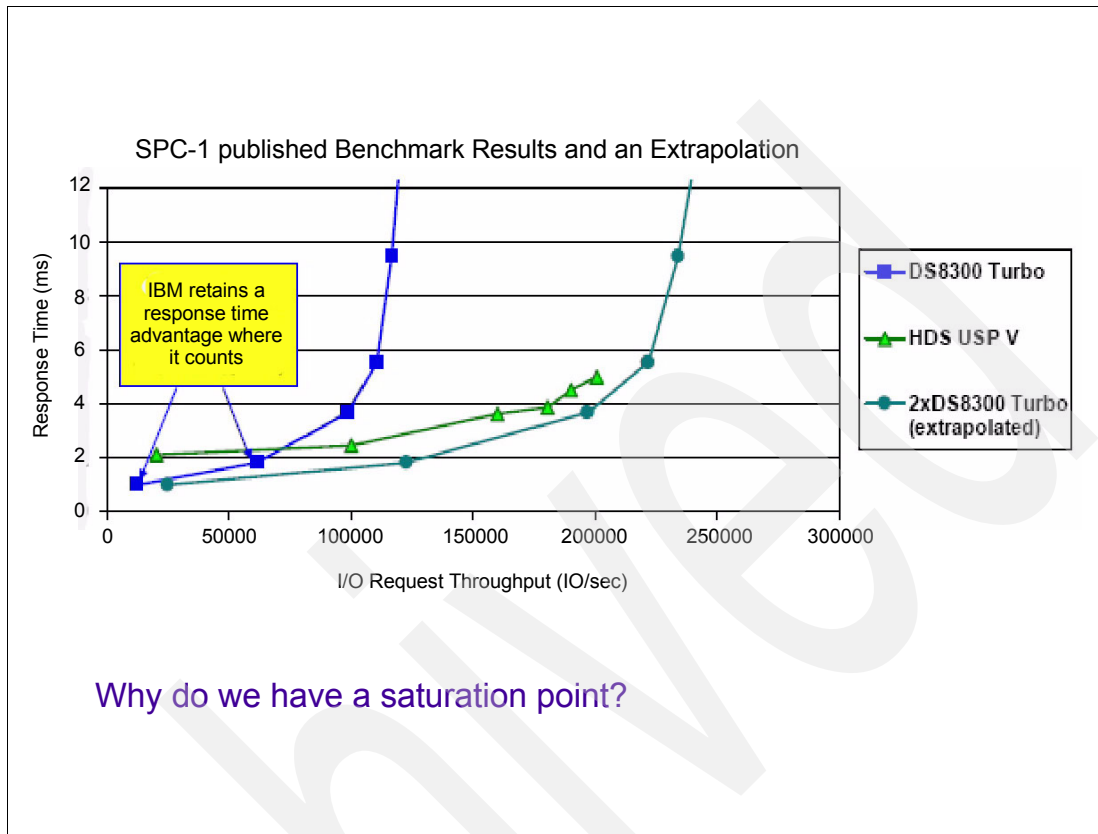


Figure 2-66 DASD saturation curve

DASD saturation curve

The well-known shaped curve as depicted in Figure 2-66 on page 168 is a direct consequence of Markov's Law.

The y-axis shows a response time where one of its components is T_w .

The x-axis shows the I/O rate. The higher the I/O rate, the higher the utilization. Therefore, in a sense this curve is a repetition of the Markov's curve that relates T_w with utilization.

2.67 Pareto's 80/20 Rule

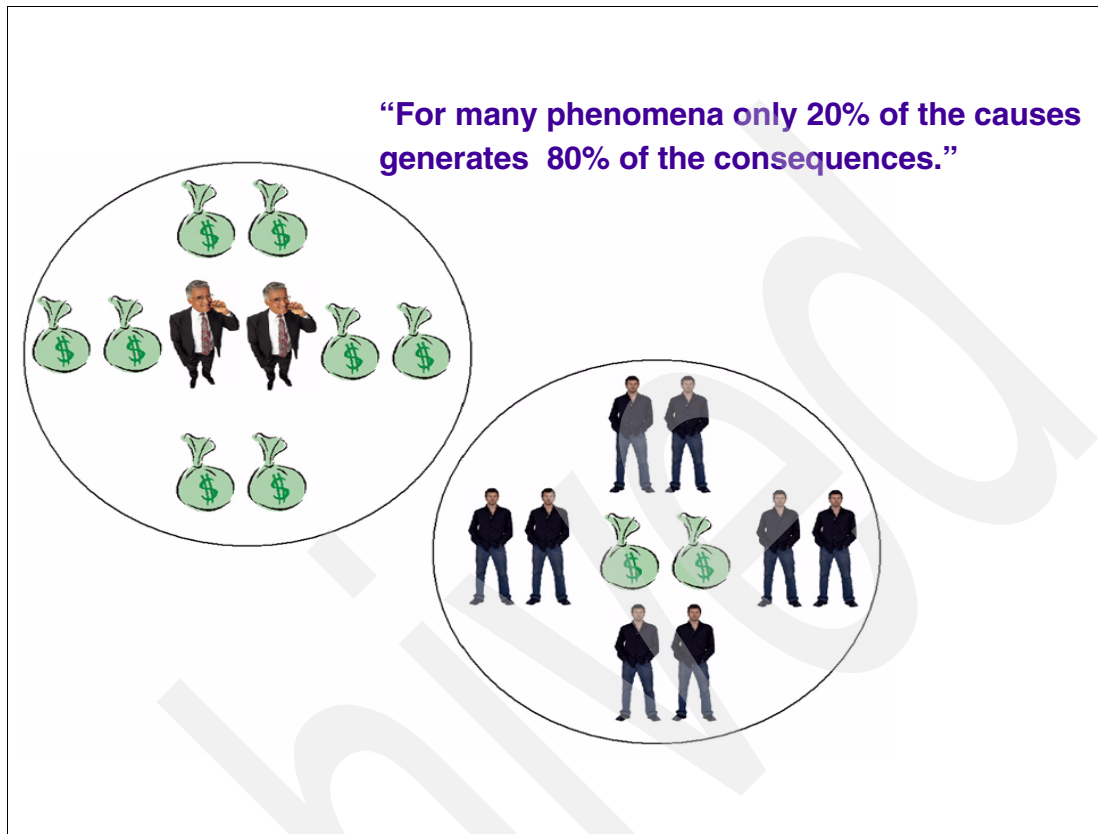


Figure 2-67 The Pareto's distribution of the economy in a society

Pareto's 80/20 Rule

The Vilfredo Pareto rule says “For many phenomena only 20% of the causes generates 80% of the consequences.”

The application of this rule to commercial data processing produces 80% of transactions consume 20% of resources, and 20% of transactions consume 80% of resources. The rule implies that trivial transactions (the majority) are responsible for 20% of consumption. A major consequence is the concept of prioritizing trivial online transactions. This aids performance in general for the following reasons:

- ▶ There is low burning in the system due to low system consumption.
- ▶ 80% of the user population is satisfied.
- ▶ It gets rid of online transactions and frees z/OS of control block manipulation, thereby saving CPU cycles.

2.68 80/20 rule and service class periods

RMF Workload Activity extract - SC TSO 1st period									
REPORT BY: POLICY=WLMPL01		WORKLOAD=TSO		SERVICE CLASS=TSO4 CRITICAL =NONE			RESOURCE GROUP=*NONE		
TRANSACTIONS	TRANS.-TIME	HHH.MM.SS.TTT	--DASD I/O--	---SERVICE----	--SERVICE RATES--				
AVG	2.54	ACTUAL	295	SSCHRT	85.0	IOC	81986	ABSRPTN	446
MPL	2.51	EXECUTION	295	RESP	40711K	CPU	1377K	TRX SERV	441
ENDED	12499	QUEUED	0	CONN	40711K	MSO	509415	TCB	152.0
END/S	6.95	R/S AFFINITY	0	DISC	0.4	SRB	66581	SRB	7.3
#SWAPS	11964	INELIGIBLE	0	Q+PEND	0.5	TOT	2035K	RCT	12.2
EXCTD	0	CONVERSION	0	IOSQ	0.9	/SEC	1132	IIT	1.9
AVG ENC	0.00	STD DEV	9.968					HST	0.0
REM ENC	0.00							APPL %	9.5
MS ENC	0.00								

RMF Workload Activity extract - SC TSO all periods									
TRANSACTIONS	TRANS.-TIME	HHH.MM.SS.TTT	--DASD I/O--	---SERVICE----	--SERVICE RATES--				
AVG	4.63	ACTUAL	524	SSCHRT	260.8	IOC	323918	ABSRPTN	495
MPL	4.60	EXECUTION	516	RESP	776372M	CPU	2736K	TRX SERV	493
ENDED	15063	QUEUED	8	CONN	776372M	MSO	921281	TCB	302.0
END/S	8.38	R/S AFFINITY	0	DISC	0.4	SRB	166698	SRB	18.1
#SWAPS	14266	INELIGIBLE	0	Q+PEND	0.5	TOT	4148K	RCT	12.7
EXCTD	0	CONVERSION	0	IOSQ	1.7	/SEC	2307	IIT	5.6
AVG ENC	0.00	STD DEV	14.175					HST	0.0
REM ENC	0.00							APPL %	18.6
MS ENC	0.00								

Figure 2-68 RMF workload activity report - extract for SC TSO

80/20 rule and service classes periods

Most often, when a transaction starts, you do not know if it is trivial or not. Therefore, the concept of *periods* is utilized in a Service Class (SC) to prioritize trivial transactions.

The existence of periods in a Service Class is used to distinguish trivial resource consumption transactions from heavier transactions. All transactions in such a Service Class always start running in the first period (the trivial and the heaviest). In this period, you should define a higher importance and a more difficult goal to be achieved.

Each non-last period has the keyword *duration* (DUR), which identifies the number of service units consumed in this period by a transaction before being moved to the next period. If this number is chosen correctly, all trivial (80% of the total) transactions should finish in the first period and enjoy a high priority. The others, when reaching the duration limit, are migrated to the next period, where the priorities (generated by value of the goal and importance) are not so high.

2.69 Partition's Law

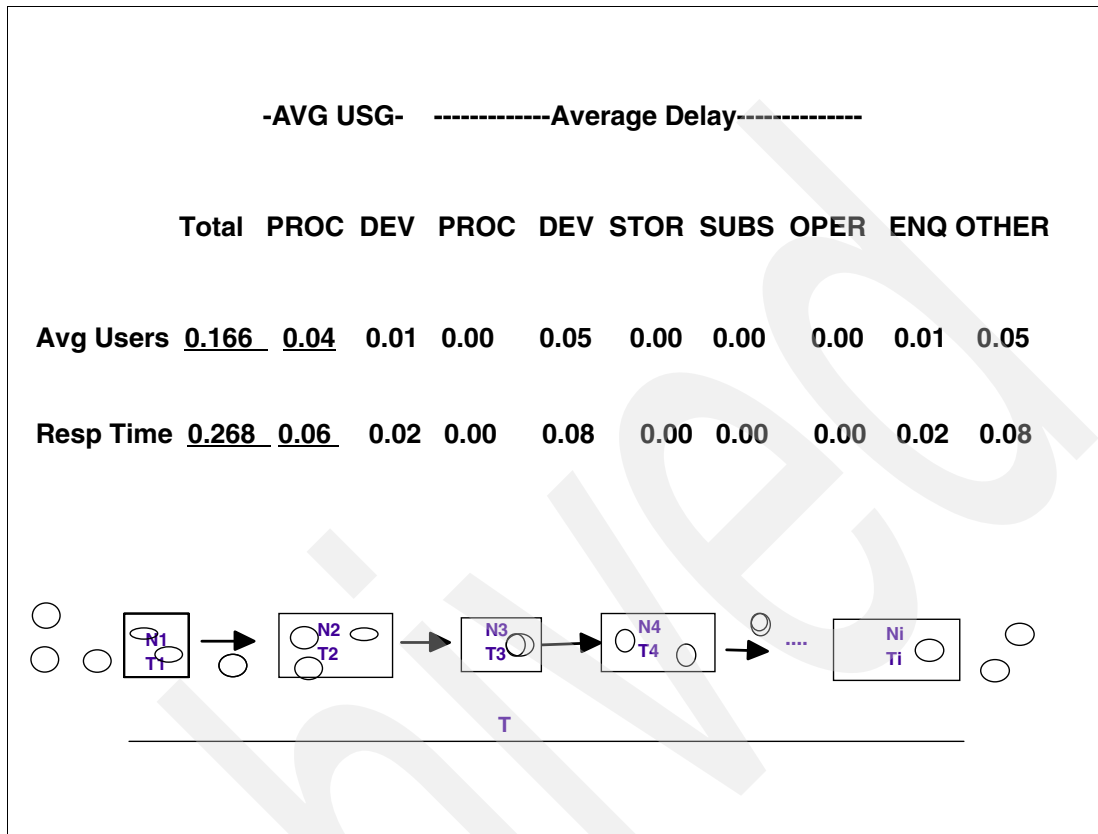


Figure 2-69 RMF Group report extract

Partition's Law

Imagine a process formed by serial stages, where transactions are processed as in a pipeline. Figure 2-69 on page 171 displays i stages, and the circles represent transactions going through the stages.

N_i is the average number of transactions being executed in the i stage. T_i is the average consumed in the i stage. If the flow of transactions is in steady state, Partition's Law states:

$$T_i = (N_i / N) * T$$

Where:

- N** Average total population in the full process (all stages)
- T** Average total time in the full process (all stages)

Using this law, you can derive each partial time in each stage (which is an important figure regarding performance), by knowing only the distribution of the population.

In Figure 2-69 on page 171, the extract of the Group RMF report shows a partition of a transaction response time. The time in each stage is calculated from the sampling of the average population in that stage. For example, for the pictured Service Class figures:

N (average number of active transactions) is 0.166.

T (average response time) is 0.268.

Then, the average time spent by those transactions executing CPU is:

$$0.04 / 0.166 * 0.268 = 0,06 \text{ msec}$$

In a practical application, if transaction average response time is unsatisfactory and you see from the report that a major contributor to the response time is I/O delay, you can direct your efforts to improving the I/O.

2.70 z10 EC SRM Constants

Table 3-3. IBM System z10 Enterprise Class (z10 EC) Processor Models		
System z10 EC Processor Model	Service Units Per Second of Task or SRB Execution Time	Seconds Task or SRB Execution Time Per Service Unit
z10 EC, Model 401	11291.4608	0.000089
z10 EC, Model 402	10680.9079	0.000094
z10 EC, Model 403	10315.9252	0.000097
z10 EC, Model 404	10050.2513	0.000100
z10 EC, Model 405	9846.1538	0.000102
z10 EC, Model 406	9673.5187	0.000103
z10 EC, Model 407	9518.1440	0.000105
z10 EC, Model 408	9373.1693	0.000107
z10 EC, Model 409	9243.2120	0.000108
z10 EC, Model 410	9111.6173	0.000110
z10 EC, Model 411	8973.6399	0.000111
z10 EC, Model 412	8834.8978	0.000113
z10 EC, Model 501	24427.4809	0.000041
z10 EC, Model 502	23021.5827	0.000043
z10 EC, Model 503	22222.2222	0.000045
z10 EC, Model 504	21621.6216	0.000046
z10 EC, Model 505	21136.0634	0.000047
z10 EC, Model 506	20725.3886	0.000048
z10 EC, Model 507	20356.2341	0.000049

Figure 2-70 z10 EC SRM constant table

Law of Diminished Returns and SRM constants

Figure 2-70 on page 173 shows the top of the table of SRM constants for z10 EC models. The table lists the service consumed per second of execution time by CPU model. The values listed are SRM constants. The total system absorption rate reported by RMF will not equal the values listed here because these do not include certain types of system processing.

SRM constant changes

Certain OPT parameters make it more convenient for installations with unique resource management requirements to change some SRM constants. The defaults provided are adequate for most installations. A parameter needs to be specified only when its default is found to be unsuitable for a particular system environment.

The SRM invocation interval control parameter in the IEAOPTxx parmlib member can be modified. This parameter, specified by the RMPTTOM keyword, controls the invocation interval for SRM timed algorithms. Increasing this parameter above the default value reduces the overhead caused by SRM algorithms, such as swap analysis and time slicing. However, when these algorithms are invoked at a rate less than the default, the accuracy of the data on which SRM decisions are made, and thus the decisions themselves, might be affected.

In the table shown in Figure 2-70 on page 173, the second column (Service Units per Second of task or SRB Execution Time) depicts the SRM constant. This metric measures the speed (in SUs per sec) of each processor in different CECs. In the example of z990 models, notice that CPU speed decays when the number of CPUs increases.

2.71 Principle of Locality

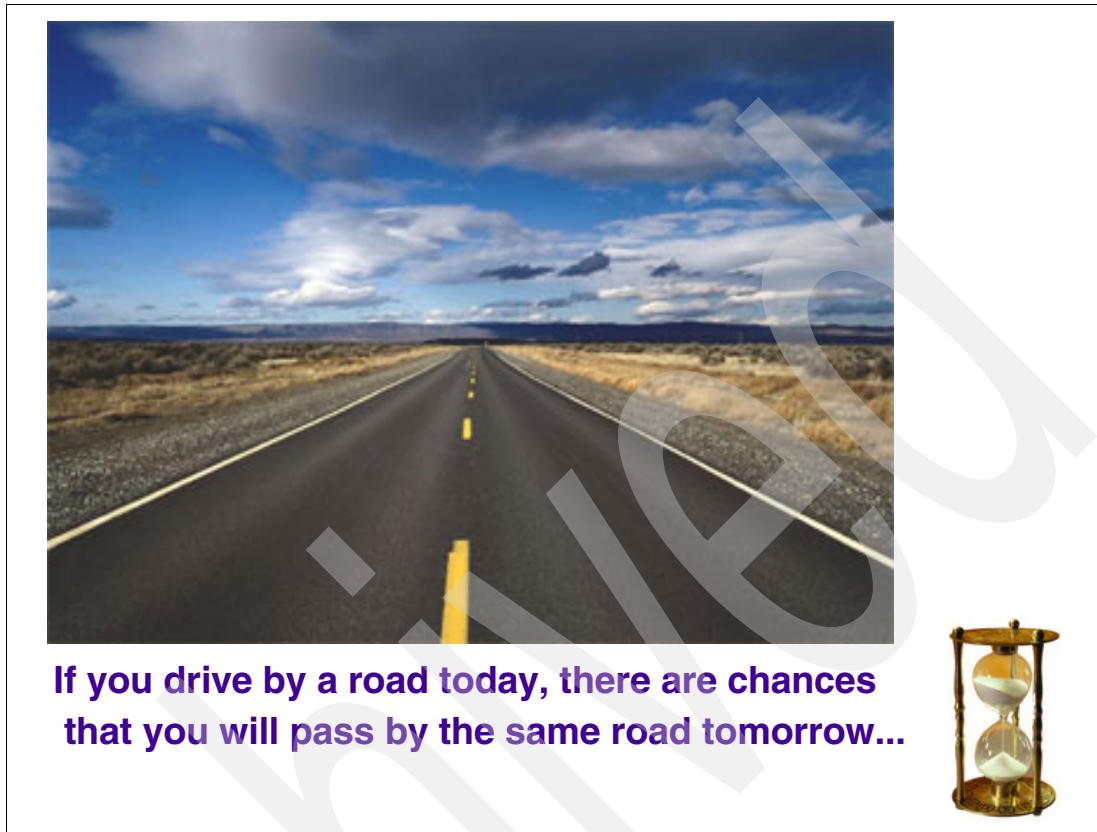


Figure 2-71 Principle of Locality

Principle of Locality

The Principle of Locality is valid in a computational commercial environment: “If a fact happens right now, there is a good chance that it will also happen in the near future.”

Consider the following examples:

► Instructions

About 20% of program code is executed frequently; that is, it is a sort of kernel containing the major logic to be applied to each I/O record. The remaining instructions are in the program simply to handle exceptional events. Therefore, if one instruction is executed now, there is a good chance that it belongs to the kernel and it will be executed again, soon.

► Data

If bank customers access their personal data now, there is a good chance that the same data will be accessed in the near future.

This behavior allows the implementation of the least recently used (LRU) algorithm, which keeps in memory the piece of code and data most referenced. This memory could be, for example, CEC Central storage (making feasible virtual storage), DASD cache, or a DB2 buffer pool.

However, there are types of processing (for example, I/O sequential processing) to which the Principle of Locality does not apply, and consequently, the same happens with the LRU algorithm. In I/O sequential operations, every processed record is not processed anymore.

Resource Measurement Facility

Resource Measurement Facility (RMF) is a product that produces reports about problems as they occur, so that an installation can take action before they become critical. This chapter describes how users can interpret the information provided by RMF.

This chapter describes the following topics:

- ▶ RMF Monitor I, II, and III
- ▶ RMF Monitor III Contention Analysis
- ▶ The workflow concept in content analysis
- ▶ Proc and DEV workflow%
- ▶ RMF Postprocessor
- ▶ RMF Spreadsheet Reporter
- ▶ RMF Performance Monitoring (RMF PM)

Also described are SMF records, including explanations of their importance to your installation, what records are necessary, and how to manage them. 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.

3.1 Resource Measurement Facility overview

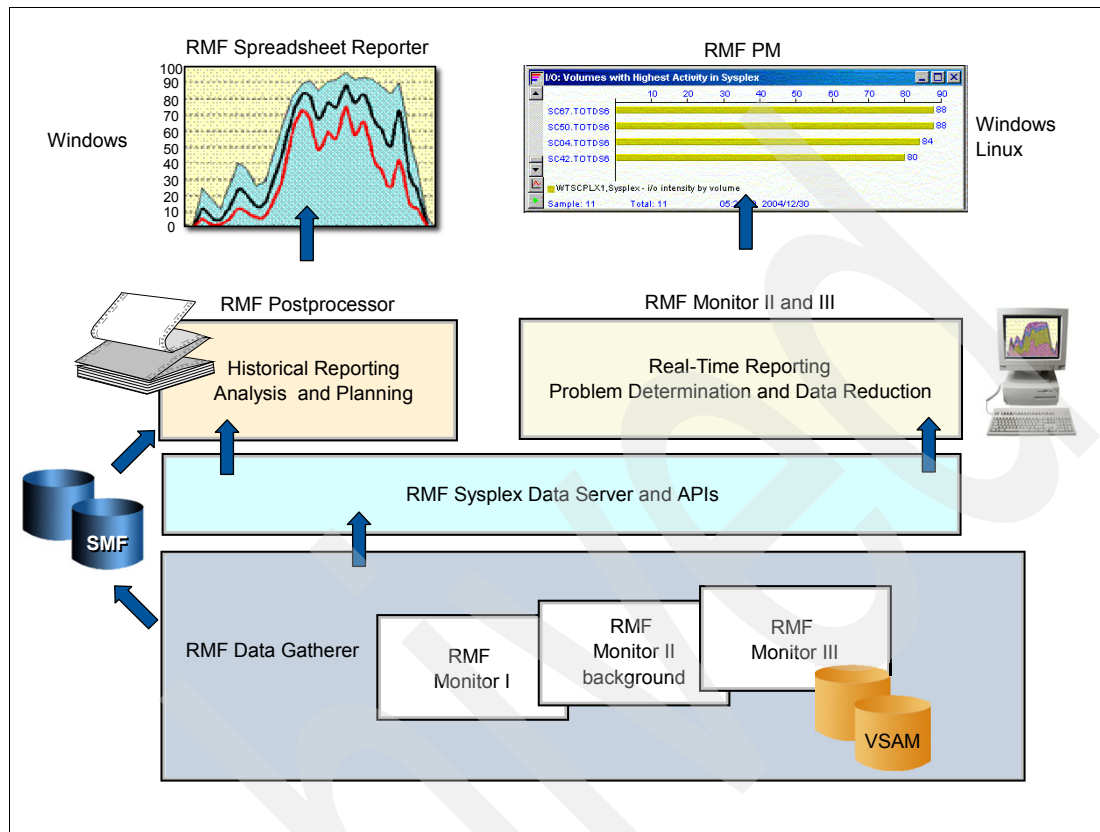


Figure 3-1 RMF product overview

Resource Measurement Facility

RMF is the strategic IBM product for performance analysis, capacity planning, and problem determination in a z/OS host environment.

Many different activities are required to keep your system running smoothly, and to provide the best service on the basis of the available resources and workload requirements. This work is done by system operators, administrators, programmers, or performance analysts.

RMF produces reports about problems as they occur, so that an installation can take action before the problems become critical. An installation can use RMF to do the following:

- ▶ Determine that a system is running smoothly
- ▶ Detect system bottlenecks caused by resource contention
- ▶ Evaluate the service that an installation provides to various groups of users
- ▶ Identify the workload delayed, and the reason for the delay
- ▶ Monitor system failures, system stalls, and failures of selected applications

For more information about RMF, visit the RMF home page:

<http://www.ibm.com/servers/eserver/zseries/zos/rmf>

RMF monitors

RMF comes with three monitors, namely Monitor I, Monitor II, and III. Because Monitor III has the ability to determine the cause of delay, use it to start your system tuning activities.

Monitor III

Monitor III provides short-term data collection and online reports for continuous monitoring of system status and solving performance problems. Monitor III is useful to begin system tuning because it allows the system tuner to distinguish between delays for important jobs and delays for jobs that are not as important to overall system performance.

Monitor I

Monitor I provides long-term data collection for system workload and resource utilization. The Monitor I session is continuous, and measures various areas of system activity over a long period of time.

You can obtain Monitor I reports directly as real-time reports for each completed interval (single-system reports only), or you can let the Postprocessor run to create the reports, either as single-system reports or as sysplex reports. Many installations produce daily reports of RMF data for ongoing performance management. In this publication, sometimes a report is called a Monitor I report (for example, the Workload Activity report), although it can be created only by the Postprocessor.

Monitor II

Monitor II provides online measurements on demand for use in solving immediate problems. A Monitor II session can be regarded as a snapshot session. Unlike the continuous Monitor I session, a Monitor II session generates a requested report from a single data sample. Because Monitor II is an ISPF application, you can use Monitor II and Monitor III simultaneously in split-screen mode to get different views of the performance of your system.

In addition, you can use the RMF Spreadsheet Reporter to further process the measurement data on a workstation with the help of spreadsheet applications. The following sections provide sample reports, including the name of the corresponding macro.

3.2 RMF Performance Management panel

```
ERB0PRM                RMF - Performance Management                z/OS V1R11 RMF
Selection ==>

Enter selection number or command on selection line.

 1 Postprocessor      Postprocessor reports for Monitor I, II, and III      (PP)
 2 Monitor II        Snapshot reporting with Monitor II                (M2)
 3 Monitor III       Interactive performance analysis with Monitor III    (M3)

U USER              User-written applications (add your own ...)      (US)

R RMF SR            Performance analysis with the Spreadsheet Reporter
P RMF PM            RMF PM Java Edition
N News              What's new in z/OS V1R11 RMF

                        T TUTORIAL      X EXIT

RMF Home Page:      http://www.ibm.com/servers/eserver/zseries/rmf/

5694-A01 Copyright IBM Corp. 1994, 2009. All Rights Reserved
Licensed Materials - Property of IBM
```

Figure 3-2 ISPF RMF primary menu

RMF primary panel

The RMF Performance Management menu offers you easy access to the reporting capabilities of the Monitor II and Monitor III display sessions and the Postprocessor. Enter the TSO/E command RMF and then you can access RMF through an ISPF command line interface, as shown in Figure 3-2. This is the RMF primary panel.

RMF provides a significant amount of information for CP utilization purposes, including CPU, I/O, and memory. It correlates resources with products such as Lotus® Domino®, WebSphere, and so on. Questions about VSAM RLS, Lotus Domino, HTTP, UNIX System Services, Fibre Channel activity, HiperSockets™ and Coupling Facility activity are frequently addressed.

RMF has been integrated with several products and z/OS features. The traditional RMF reporting is complemented by powerful workstation features that provide a state-of-the-art graphical user interface. Tivoli® integration is important because it enables clients to build an enterprise-class performance monitoring infrastructure.

Note: Online help for RMF reporting sessions includes a tutorial, help, and message help panels. To obtain more information about a report or a panel, press PF1. To use the RMF tutorial, either enter the T command on the Primary menu, or enter =T (using the ISPF “jump” facility) from the command line on any panel.

3.3 RMF monitors

- ❑ RMF has three data gatherers/reporters known as monitors:
 - Monitor I - Long-term data collection
 - Monitor II - Online snapshot monitoring
 - Monitor III - Online short-term and long-term data gathering
- ❑ RMF Spreadsheet Reporter
 - Measurement data on a workstation

Figure 3-3 RMF monitors

RMF monitors

For the disciplines mentioned in 3.1, “Resource Measurement Facility overview” on page 176, different kinds of data collectors (gatherers) and reporters are needed. As mentioned, RMF comes with three monitors known as Monitor I, Monitor II, and Monitor III.

All three monitors write SMF records (type 70 - type 79) if you define the appropriate SMF recording options. In addition, Monitor III writes VSAM records to in-storage buffers or into RMF-owned VSAM data sets. RMF also provides the RMF Postprocessor program, which reads SMF records generated by the monitors producing sysout reports.

RMF Spreadsheet Reporter overview

The RMF Spreadsheet Reporter is a powerful workstation solution for graphical presentation of RMF Postprocessor data. Use it to convert your RMF data to spreadsheet format and generate representative charts for all performance-relevant areas.

Performance data derived from SMF records is the basis for z/OS performance analysis and capacity planning. The basic idea of the RMF Spreadsheet Reporter is to exploit the graphical presentation facilities of a workstation for these purposes: it extracts performance measurements from SMF records; produces Postprocessor Report Listings and Overview Records; and converts this postprocessor output into spreadsheets. Thus, the Spreadsheet Reporter offers a complete solution of enhanced graphical presentation of RMF measurement data.

The Spreadsheet Reporter also provides several sample spreadsheet macros to help you in viewing and analyzing performance data at a glance.

For more detailed information, refer to “RMF Spreadsheet Reporter” on page 194. For a description of how to use these functions, refer to *z/OS Resource Measurement Facility User's Guide*, SC33-7990.

Archived

3.4 RMF Monitor I

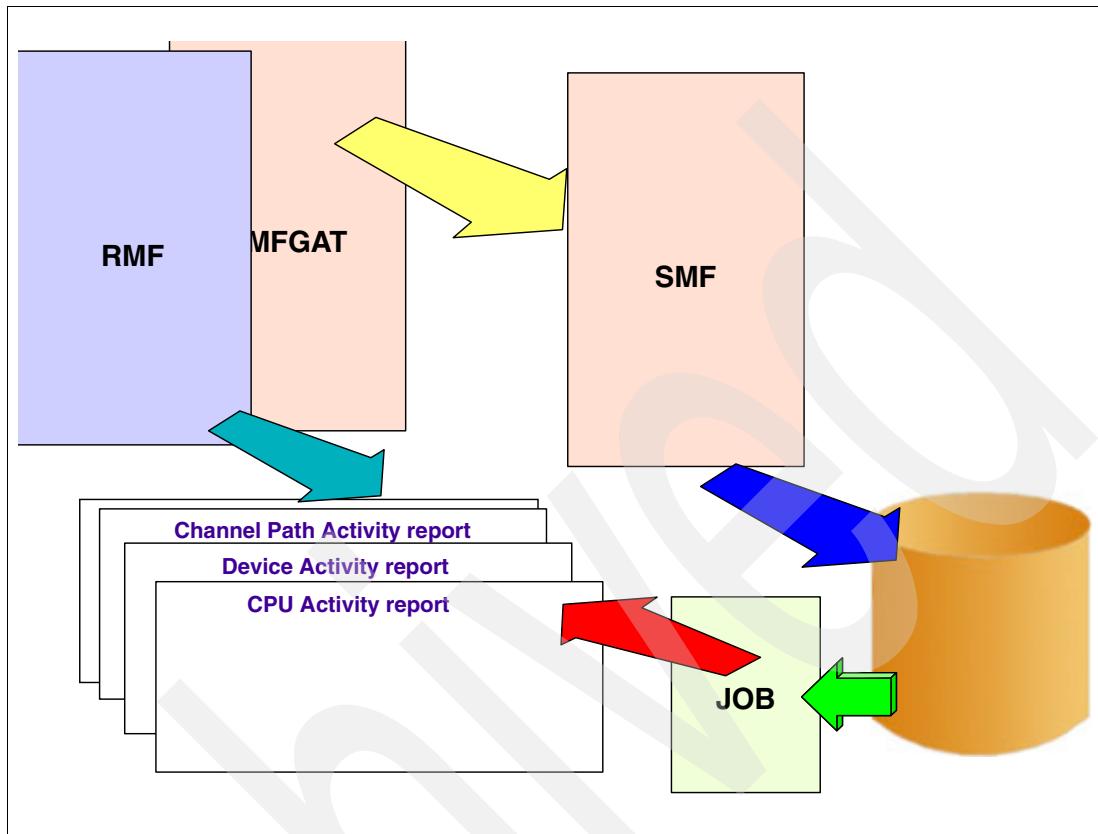


Figure 3-4 RMF Monitor I

RMF Monitor I

RMF Monitor 1 provides long-term data collection for system workload and system resource utilization. The Monitor I session is continuous, and measures various areas of system activity over a long period of time.

You can obtain Monitor I reports directly as real-time reports for each completed interval (single-system reports only). Or you can allow the RMF Postprocessor program to create the reports, either as single-system or sysplex reports. Many installations produce daily reports of RMF data for ongoing performance management.

Monitor I has two types of components: a *data gatherer* that records SMF records, and a *data reporter* that reads these records in memory and generates real-time reports. The SMF data produced can also be processed by the RMF Postprocessor.

The SMF data collected by Monitor I is mostly used for capacity planning, but is also used for performance analysis. Additionally, a few products, such as Tivoli Decision Support, use SMF records gathered by Monitor I.

Note: For Monitor I, use the default sampling cycle of 1 second. For the reporting interval, the default of 30 minutes is fine to start with. Adjust this if you prefer, or if you need to focus in on a problem.

3.5 RMF Monitor II

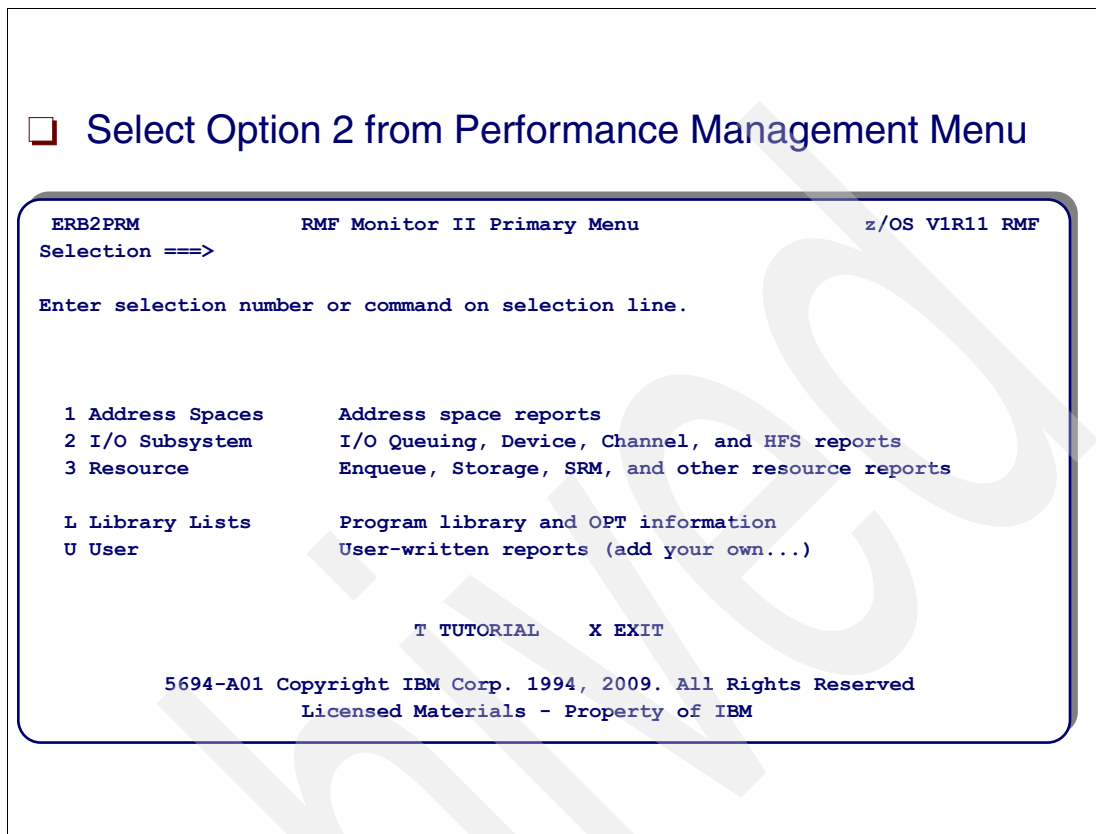


Figure 3-5 TSO RMFMON menu

RMF Monitor II

RMF Monitor II provides online measurements on demand for use in solving immediate problems. A Monitor II session can be regarded as a snapshot session collecting data about address space states and resource usage. Unlike the continuous Monitor I session, a Monitor II session generates a requested report from a single data sample. With Monitor II, it is also possible to monitor one specific job or volume continuously.

The scope of Monitor II data gathering is mainly related to single address spaces or resources, giving snapshots of the current status. You can collect data about address space activities and resource consumption, and about processor, DASD volume, and storage activities and utilization.

RMF Monitor II data gatherer and data report components are together in Monitor II. Because Monitor II is an ISPF application, you can use Monitor II and Monitor III simultaneously in split-screen mode to get different views of the performance of your system. To invoke the RMF Monitor II session, you can do either of the following:

- ▶ Select the **Monitor II** option from the Performance Management menu. To return to the menu, issue the **M** command.
- ▶ Or issue the **TSO RMFMON** command. To exit RMFMON, issue the **Z** command.

Certain reports offer continuous monitoring of single address spaces or DASD devices. You can get a one-line report each time you press Enter, or you can request a periodically refreshed report. Figure 3-5 displays the RMF II Primary Menu.

3.6 RMF Monitor II ARD report

```

ERB2BUF          RMF - ARD Address Space Resource Data          Line 1 of 65
Command ==>                                           Scroll ==> PAGE

                CPU=  4/  4 UIC= 65K PR=  0                System= SC70 Total

12:11:08 DEV   FF   FF PRIV LSQA X C SRM  TCB   CPU  EXCP SWAP LPA CSA NVI V&H
JOBNAME  CONN 16M  2G   FF  CSF M R ABS  TIME  TIME RATE RATE  RT  RT  RT  RT

*MASTER* 3410  0  224 4136 1986      0.0 471.4 2237  -----
PCAUTH  0.003  0   65   4  108 X  0.0  0.00  0.00  -----
RASP    0.000  0   38  486   58 X  0.0  0.00 14.03  -----
TRACE   0.034  0   41 1551   83 X  0.0  0.00  0.00  -----
DUMPSRV 5.091  0   44   4  153      0.0  0.68  0.81  -----
XCFAS   13440  0 2232 3049 7050 X  0.0 1898 2742  -----
GRS     448.9  0  705 1333 18K X  0.0 674.6 1439  -----
SMSPDSE 0.055  0   47  317 159 X  0.0 60.54 73.21  -----
SMSPDSE1 0.469  0   49  833 200 X  0.0 70.73 83.69  -----
SMSVSAM 3.479  0  199  352 18K X  0.0 1471 2337  -----
CONSOLE 0.754  0   31   69 139 X  0.0 61.02 69.71  -----
WLM     0.065  0   82   57 288 X  0.0 9257 10687  -----
ANTMAIN 1.257  0   29   6  214 X  0.0 13.90 16.79  -----
ANTAS000 1.266  0   33   6  180 X  0.0  0.47  0.51  -----
DEVMAN  0.117  0   21   6   71 X  0.0  1.07  1.73  -----
OMVS    88.15  0  213  253 301 X  0.0 64.98 79.70  -----
IEFSCHAS 0.032  0    3   4   42 X  0.0  0.00  1.04  -----
JESXCF  1.474  0   25   11 103 X  0.0 506.7 870.5  -----

```

Figure 3-6 RMF Monitor II ARD report

RMF Monitor II ARD report

You can display the report shown in Figure 3-6 by selecting Option 1, shown in Figure 3-5 on page 182. When the next panel appears, select Option 1 again. Figure 3-6 shows an example of the ARD report, which is one of the Monitor II reports. The ARD report provides information about the system resources that are used by each address space in the system, or by each address space that meets the selection criteria that you specify when you request the report.

This report enables you to determine which jobs are creating performance problems. After a problem job has been identified, you can request an ARDJ report for that particular job. This enables you to focus your reporting on a known problem area.

The report shows data such as processor time, paging, and main storage.

- ▶ In the X M column, X indicates that the address space is accessed from other address spaces by means of cross-memory functions.
- ▶ The C R column indicates if WLM managed the address space as CPU-critical (C) or storage-critical (S) or both (X) during the reporting interval.

For more information about this topic, see *z/OS Resource Measurement Facility Report Analysis*, SC33-7991.

3.7 RMF Monitor III primary panel

```
ERB3PRM                RMF Monitor III Primary Menu                z/OS V1R11 RMF
Selection ==>

Enter selection number or command on selection line.

S SYSPLEX              Sysplex reports and Data Index              (SP)
1 OVERVIEW             WFEX, SYSINFO, and Detail reports              (OV)
2 JOBS                 All information about job delays              (JS)
3 RESOURCE             Processor, Device, Enqueue, and Storage              (RS)
4 SUBS                 Subsystem information for HSM, JES, and XCF              (SUB)

U USER                User-written reports (add your own ...)              (US)

O OPTIONS              T TUTORIAL              X EXIT

5694-A01 Copyright IBM Corp. 1986, 2009. All Rights Reserved
Licensed Materials - Property of IBM
```

Figure 3-7 Using RMF Monitor III

RMF Monitor III

To select the RMF Monitor III panel, select Option **3** as shown in Figure 3-2 on page 178. RMF Monitor III provides short-term data collection and online reports for continuous monitoring of system status and for solving performance problems, workflow delay monitoring, and goal attainment supervision. Monitor III is a useful place to begin tuning; it can sometimes help to determine the cause of a delay before the delay is noticed by users.

RMF Monitor III data is more transaction-oriented than system-oriented. The Monitor III gatherer session has a typical gathering cycle of one second, and consolidated VSAM records are written for a range that is typically set to 100 seconds. You can collect short-term data and continuously monitor the system to solve performance problems. You get actual performance data (response times, execution velocity) on a detailed level for later comparison with performance policy goals. This simplifies the comparison of reports created from Monitor I and Monitor III data.

Using the Monitor III primary panel

To start a Monitor III session, enter the TSO/E command **RMF**. Then select **Monitor III** from the “RMF - Performance Management” panel that comes up and select Option **3** as shown in Figure 3-2 on page 178. The panel that RMF displays in response to your selection is shown in Figure 3-7.

The selection line options are as follows:

- S SYSPLEX** This command displays the Sysplex Report Selection menu. Use this menu to select one of the sysplex reports, or the Data Index.
- 1 OVERVIEW** This command displays the Overview Report Selection menu. Use this menu to select Workflow/exceptions, system information, and various detail reports.
- 2 JOBS** This command displays the Job Report Selection menu, which shows available reports about job delays. Use this menu to choose the specific job you want to analyze and the type of delay you want reported.
- 3 RESOURCE** This command displays the Resource Report Selection menu. Use this menu to select reports on processors, devices, enqueue and storage. Use this menu to choose the resource for which you want to see delays or storage problems.
- 4 SUBS** This command displays the Subsystem Report Selection menu. Use this menu to select HSM, JES, and XCF Delay reports.
- U USER** This command displays the User-written Report Selection Menu. Use this menu to select your user-written reports or those examples that are provided with Monitor III.

Monitor III components

RMF Monitor III has two types of components:

► Data gatherer (RMFGAT)

Preparation of data gathering with Monitor III requires the following steps:

- Defining VSAM data sets
- Ensuring common storage tracking

IBM provides the cataloged procedure needed to start the Monitor III gatherer session. It is stored in SYS1.PROCLIB(RMFGAT):

```
//IEFPROC EXEC PGM=ERB3GMFC,REGION=128M,TIME=1440
```

► Data reporter

To display a user-modified or user-created report, RMF uses 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.

A Monitor III Reporter session that uses preallocated data sets does not require the Monitor III data gatherer to be running on the same system. You can therefore display, on one system, the data that RMF has gathered on another system. This allows you for example to run Monitor III Reporter sessions on one system, and send the data sets from other locations to be analyzed there. After transmission, the data sets can be preallocated and then analyzed during a reporter session in the usual manner.

The Monitor III data reporter runs in a TSO/E session under ISPF. The data reporter reads VSAM data generating window reports providing sysplex or system performance information such as:

- Displaying your current system status in real-time mode
- Showing previously collected data that is still available in either in-storage buffers or preallocated VSAM data sets

3.8 RMF Monitor III contention analysis

- Each address space or enclave may have the states:
 - Using
 - Delay
 - Idle
 - Unknown

```

RMF V1R11 Delay Report
Command ==>
Samples: 120
System: MVS1 Date: 11/28/09 Time: 12.00.00
Service MFL USG DLY IDL UKN ---- % Delayed For ---- Primary
Name CX Class Cr % % % % PRC DEV STR SUB OPR ENQ Reason
*SYSTEM 49 1 1 62 36 0 0 0 0 0 0 0 0
*TSO 56 1 1 95 2 0 0 0 1 0 0 0
*BATCH 39 2 4 0 94 1 0 0 0 4 0 0 0
*STC 40 0 1 51 48 0 0 0 0 0 0 0 0
*ASCH 0 0 0 0 0 0 0 0 0 0 0 0 0
*OMVS 0 0 0 0 100 0 0 0 0 0 0 0 0
*ENCLAVE 0 0 0 0 0 0 N/A 0 N/A N/A N/A
JES2 S SYSSTC 0 0 1 0 99 0 1 0 0 0 0 SCLSP4
BMAI T PRDTSO S 15 9 66 13 4 0 1 0 65 0 0 HSM
HSM S STCCMD C 30 26 62 0 23 0 1 0 0 50 11 Mount
HIRW2 B BATCHMED 35 6 14 0 1 2 1 0 12 0 0 HSM
TCPNET SO SYSSTC SC 60 3 2 0 97 2 0 0 0 0 0 NET
*MASTER* S SYSTEM 67 3 2 0 95 0 2 0 0 0 0 M00202
  
```

Figure 3-8 Address space and enclave states

Contention analysis

The Delay report allows you to determine which system resources are causing delays for jobs or job groups, and to what extent the jobs are delayed.

Primary reason for delay

Reported only for a specific job, this field provides additional information about the primary reason for the delay. The contents depend on the resource having the largest % Delayed for value. The report gives you information about job delays for every type of delay that RMF monitors.

This includes the following delays:

- ▶ Processor delay (PRC)

This field contains the name of the job that used the processor most frequently while the reported job was delayed.

- ▶ Device delay (DEV)

This field identifies the cause of the largest percentage of delay:

- COMM common storage paging (includes shared pages)
- LOCL local storage paging (includes shared pages)
- VIO virtual I/O paging
- SWAP swap-in delay
- OUTR swapped out and ready

- XMEM cross-memory address space
- HIPR standard Hiperspace™ paging delays
- ▶ Storage delay (STR)

This field contains the volume serial number of the device that the reported job was most frequently delayed for.
- ▶ Subsystem delay (SUB)

This field contains either JES, HSM, or XCF, depending on which subsystem is causing the most delay.
- ▶ Operator delay (OPR)

This field contains Message if most of the delay was due to a message. The field contains Mount if most of the delay was due to a mount request.

The field can contain QUIESCE if the operator quiesced the address space. A quiesced address space can show unexpected data, as explained here:

 - A swappable address space will be swapped out, thus it can be OUTR and show storage delays.
 - A non-swappable address space will get lowest priority, thus it can show CPU delay, paging delay, or other delays, and even USG % from time to time, depending on the load on the system.

Cursor-sensitive control on this field gives you the Quiesce delay variation of the Job Delay report.
- ▶ Enqueue delay (ENQ)

This field contains the major name of the resource most responsible for the delay.

Note: RMF provides a detail report for each of these delays except OPR. OPR, or operator delay, includes message, mount, and quiesce requests. SUB is divided into an HSM, JES, and XCF detail report. The names of the detail reports correspond to the names that appear in the Delay report.

3.9 RMF Workflow Exceptions report

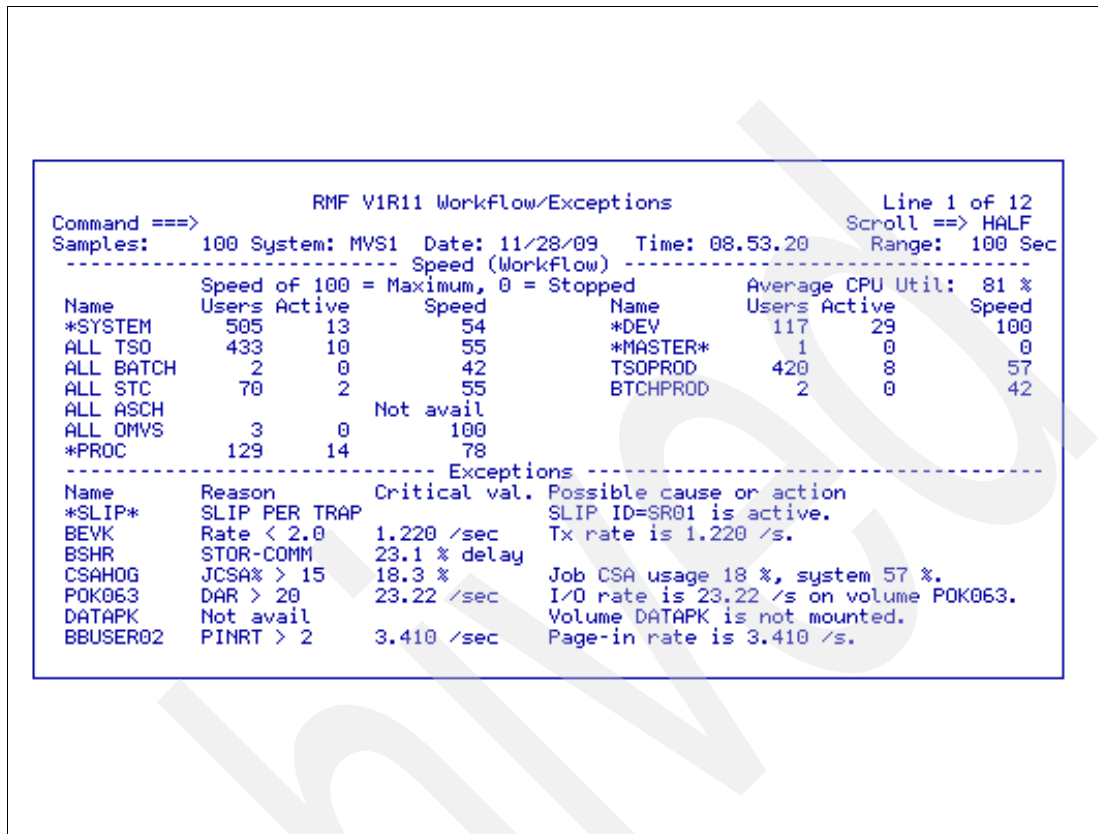


Figure 3-9 RMF Workflow Exceptions report

Workflow Exceptions report

To request the Workflow/Exceptions, select Option 1 as shown in Figure 3-7 on page 184, then select 1 from the Overview Report menu that is displayed to display or enter the following command:

```
WFEX
```

The report has two parts:

- ▶ In the Speed section, shown at the top, RMF reports the workflow of jobs and resources as speed relative to the maximum speed with which they can move through the system.
- ▶ In the Exceptions section, shown at the bottom, RMF lists jobs, job groups, or system resources that meet exception criteria.

Note: The workflow and exception lines are color coded according to severity; usually red indicates a problem, yellow indicates caution, and turquoise indicates that a job or volume is missing from the system configuration. You can specify exception criteria on the Workflow/Exceptions Report Options panels, or you can use automatic customization.

Contents of the report

Workflow of jobs or job groups is a measure of the speed at which jobs are moving through the system in relation to the maximum speed at which the jobs can move through the system.

A low workflow percentage indicates that a job has few of the resources it needs and is contending with other jobs for system resources. A high workflow percentage indicates that a job has the resources it needs to execute, and that it is moving through the system at a relatively high speed.

Example: A job that can execute in one minute if all the resources that it needs are available will have a workflow of 25% if it takes four minutes to execute.

Workflow of resources (processors or devices) represents how well the system is serving users. The speed at which each resource performs the work of all users is expressed as a value from 0% to 100%. A low resource workflow percentage represents a large queue of work requests from users. A high workflow percentage represents little resource contention.

If `Not avail` appears on your report, it means that the job you selected on the Definition and Criteria panel was not running during the report interval. If `No work` appears, the job or job group was idle (not requesting system resources) during the report interval.

Speed (workflow) section

For jobs and job groups, Speed is a measure of the speed at which jobs are moving through the system in relation to the maximum speed at which the jobs can move through the system.

A low workflow percentage indicates that a job has few of the resources it needs and is contending with other jobs for system resources. A high workflow percentage indicates that a job has the resources it needs to execute, and that it is moving through the system at a relatively high speed.

For resources (processors or devices), Speed represents how well the system is serving the users. A low resource workflow percentage represents a large queue of work requests from users. A high workflow percentage represents little resource contention.

Exceptions section of the report

See *z/OS Resource Measurement Facility Report Analysis*, SC33-7991, for a complete description of this report. Descriptions of various fields are listed here.

Name	This field contains is the one- to ten-character identifier of a workflow indicator. It can be a job, job group, or resource (processor or device). You can specify Name on the Label field of the Definition and Criteria panel or leave it blank and use the default name generated by RMF. If a threshold from the Definition and Criteria options panel is exceeded, one or more lines in the Exceptions section are shown with a name from the Label field, a specific job name, or resource name.
Reason	This field contains the explanation for the exception condition that was defined either on the WFEX Report Options panel or by automatic customization.

3.10 Definition and Criteria panel

- ❑ Use this panel to modify the report
 - By defining or changing workflow indicators and exception conditions

```
RMF WFEF Report Options: Definition and Criteria
Command ==>                               Scroll ==> HALF
Enter or edit information below. To view a list of criteria name values,
place the cursor in a blank "Name" field and press ENTER.
Exception will be displayed if all criteria of one color in a set are met.
Class ==> ----- For example: SYSTEM, BATCH, JOB, DEV, STC, SRVCLS
Qualifier ==> ----- For example: Jobname, volume serial, job class
Indicator ==> ----- WF, EX-ANY, EX-AVG, EX-GROUP or EX-UNAVAIL
Label ==> ----- Label for workflow monitor or exception line
Alert ==> ----- Alerting signal: BLINK, BEEP, BOTH, NONE
Text ==> ----- Leave blank for default

Criteria set 1          Criteria set 2          Criteria set 3
Name  < Yel  Red      Name  < Yel  Red      Name  < Yel  Red
----- -- --- ---      ----- -- --- ---      ----- -- --- ---
----- -- --- ---      ----- -- --- ---      ----- -- --- ---
----- -- --- ---      ----- -- --- ---      ----- -- --- ---
----- -- --- ---      ----- -- --- ---      ----- -- --- ---
----- -- --- ---      ----- -- --- ---      ----- -- --- ---
```

Figure 3-10 Definition and Criteria panel

Definition and Criteria panel

On this panel, you modify the report by defining or changing workflow indicators and exception conditions.

In the top half of the panel, provide information about the job or job group, or resource.

In the bottom half of the panel, fill in exception values and highlighting criteria, or choose volumes or job names. You can use cursor-sensitive control on the Name field. The corresponding Criteria Names Selection panel is displayed.

To exit this panel, you must either:

- ▶ Specify a complete workflow indicator or exception condition.
- ▶ Or use the CANCEL command to cancel any input.

Note: If your help desk staff uses the Workflow/Exceptions report as a continual system monitor, they will be alerted to exception conditions. These can be your first indicators to a problem. This depends on your having customized the window to your particular needs and thresholds. For tailoring information, see *z/OS Resource Measurement Facility (RMF) Report Analysis*, SC33-7991.

WFEX GUI

Especially in the context of exception-initiated monitoring, it is often useful to start the GUI session directly with the WFEX report. If an exception criterion on the host is met, the user on the workstation immediately gets the WFEX report that gives the reason for the notification.

To achieve this, edit the procedure ERBCSGUI and overwrite the statement:

```
stdparm = m0parm          /* ISPF primary selection menu */
with:
stdparm = wfparm          /* RMF monitor III wfex      */
```

Attention: This ERBCSGUI procedure builds the GUI command string from a given parameter or from a predefined default. It then issues the ISPSTART request for the GUI connection. It also retrieves the system name from the CVT. When the connection has been made, this system name appears in the title bar of the window.

ERBCSGUI sets up the GUI connection and issues the connection request. The GUI performs the following tasks:

- ▶ It retrieves the system name from CVT.
- ▶ It builds the GUI command string from the input parameter or from the default.
- ▶ It issues the ISPSTART command for the GUI connection.

3.11 RMF Postprocessor

- ❑ z/OS-based data reduction and reporting component for SMF data from RMF and other gatherer functions
 - SMF data resides in SMF data sets or RMF data buffer
 - Postprocessor generates readable reports and output that can be used by further postprocessing functions
 - RMF Interval and Duration reports
 - RMF Exception reports
 - RMF Overview reports
 - Postprocessor reports cover:
 - Sysplex-wide resource usage for Coupling Facility and cache control units
 - Sysplex-wide workload activity
 - System-specific reports for all system resources
 - Reports useful for problem determination
 - Activities for e-business applications

Figure 3-11 RMF postprocessor

RMF Postprocessor

RMF Postprocessor reports are the preferred media to analyze SMF data for long-term reporting. However, these are tabular report listings, and it can be cumbersome to analyze their content or retrieve exactly the required performance information. With the Spreadsheet Reporter, you can easily convert these reports to spreadsheet format, and use spreadsheet macros for graphical presentation. Graphical charts are a comfortable medium for performance analysis and also significantly improve the capability of long-term and trend reporting of RMF data.

Postprocessor reports are based on data gathered as SMF records by RMF (Monitor I, Monitor II, and Monitor III), by web servers, and by Lotus Domino servers. The report types are described in the following sections.

Interval reports

Interval reports can be created either as:

- ▶ Single-system, by using the report option `REPORT(options)`
- ▶ Sysplex-wide reports, by using the report option `SYSRPTS(options)`

For sysplex-wide reports, input data can come from an SMF data set or from the RMF Sysplex Data Server. In options, you indicate the report names to be created.

Duration reports

Duration reports are similar to interval reports for the same system activities. However, a duration report summarizes the activities of all RMF measurement intervals that fall within the duration interval (that is, the period of time covered in the duration report).

Duration reports allow you to measure your system's performance over long periods of time with a minimal amount of system overhead and a minimal volume of printed output. A duration interval is requested by the DINTV(*hhmm*) option, where *hhmm* is the duration.

Exception reports

Exception reports present a summary of the values that exceeded installation-defined thresholds over a specific period of time. RMF compares the threshold values specified in the exception control statements with the computed value in the appropriate SMF record field. If the threshold is exceeded, RMF writes a line in the exception report.

You request exception reports using the EXCEPT(*options*) statement. For a description of exception control statements, refer to *z/OS Resource Measurement Facility Report Analysis*, SC33-7991.

Overview reports

Overview reports provide an improved version of the Exception and Summary report and offer data for further processing in spreadsheet or other applications.

Plot reports

Plot reports present a graphic summary of system activity over a specified time range.

Summary reports

Summary reports provide a high-level view of system activity.

3.12 RMF Spreadsheet Reporter

- ❑ Workstation extension of the RMF Postprocessor
 - Runs on Windows ME, 2000, XP
 - Integrated in the RMF product
 - Spreadsheet Reporter application files
 - Spreadsheet macros for Lotus 1-2-3 Version 9.7 or higher and for Microsoft® Excel
 - Sample RMF report listing
- ❑ Converts postprocessor reports and overview records to spreadsheet format
- ❑ Workstation interface to the RMF Postprocessor
 - Collector generates and submits job to the host
 - Download the data to the workstation
 - Generate the working set from the downloaded data

Figure 3-12 RMF Spreadsheet Reporter

RMF Spreadsheet Reporter

The RMF Spreadsheet Reporter is the powerful workstation solution for graphical presentation of RMF Postprocessor data. Use it to convert your RMF data to spreadsheet format and generate representative charts for all performance-relevant areas.

The RMF Spreadsheet Reporter offers the following features:

- ▶ Ease of use. Manage the related resources by means of an Explorer-like GUI.
- ▶ Fast path to graphical presentation. Prepare the SMF data in a single step.
- ▶ Batch mode. Generate the input files for the spreadsheets without any GUI interaction.

Performance data derived from SMF records is the basis for z/OS performance analysis and capacity planning. The basic idea of the RMF Spreadsheet Reporter is to take advantage of the graphical presentation facilities of a workstation for these purposes: it extracts performance measurements from SMF records; produces Postprocessor Report Listings and Overview Records; and converts this postprocessor output into spreadsheets. Thus, the Spreadsheet Reporter offers a complete solution of enhanced graphical presentation of RMF measurement data.

Viewing and analyzing performance

The Spreadsheet Reporter also provides several sample spreadsheet macros to help you in viewing and analyzing performance data at a glance. The RMF Spreadsheet Reporter performs the following tasks:

- ▶ It extracts performance measurements from SMF records.
- ▶ It reduces postprocessor report listings and overview records.
- ▶ It converts this postprocessor output into spreadsheets.
- ▶ It is easy to use and manages the related resources by means of an Explorer-like GUI.
- ▶ It provides a fast path to graphical presentation (it prepares the SMF data in one step).
- ▶ It generates the input files for spreadsheets without any GUI interaction.

Overview reports and records

Overview reports and Overview records are used for long-term performance analysis. Both are generated from overview control statements that specify the performance metrics that you want to examine. The Spreadsheet Reporter provides several macros that generate these statements for you, so that you do not need to know about their syntax.

Graphical charts

RMF Postprocessor reports are the preferred media to analyze SMF data for long-term reporting. However, these are tabular report listings, and it can be cumbersome to analyze their content or retrieve exactly the required performance information. With the Spreadsheet Reporter, you can easily convert these reports to spreadsheet format and use spreadsheet macros for graphical presentation. Graphical charts are a comfortable medium for performance analysis and also significantly improve the capability of long-term and trend reporting of RMF data.

3.13 RMF Spreadsheet Reporter panels

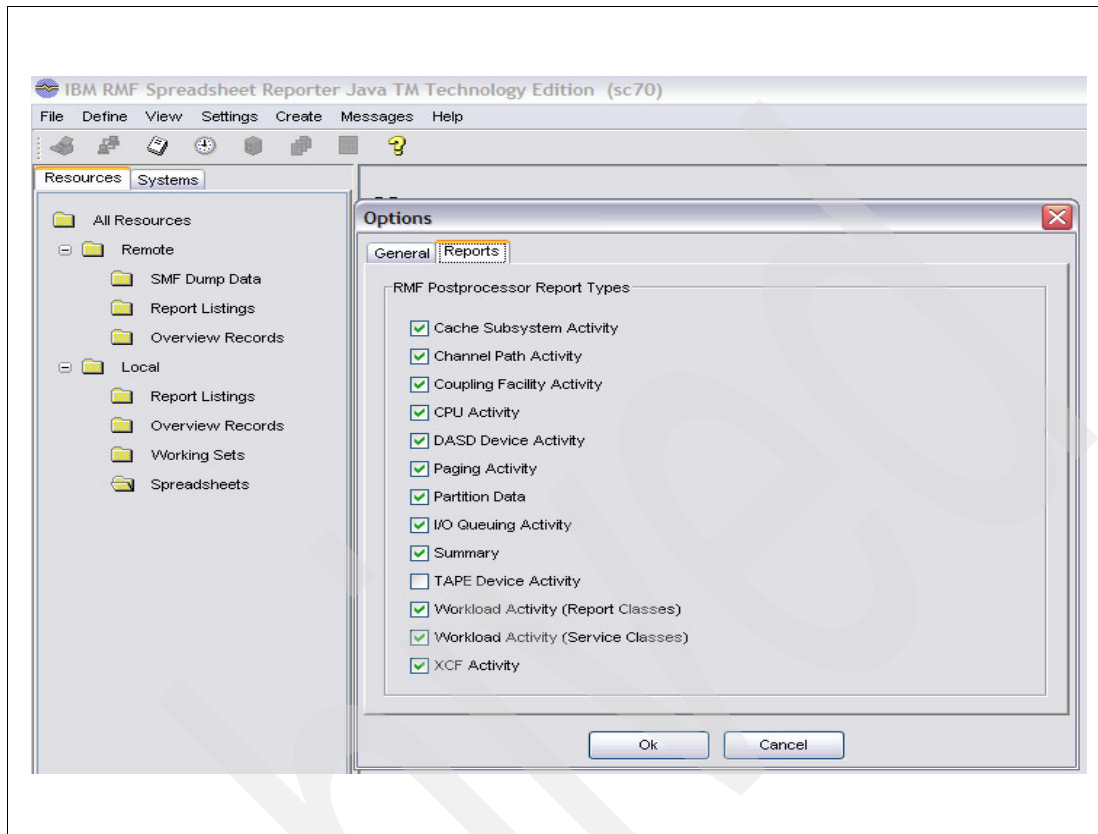


Figure 3-13 RMF Spreadsheet Reporter panel

RMF Spreadsheet Reporter panel

The RMF Spreadsheet Reporter is the powerful workstation solution used for the graphical presentation of RMF Postprocessor data. Use it to convert your RMF data to spreadsheet format and generate representative charts for all performance-relevant areas.

Figure 3-13 shows the primary panel in the RMF Spreadsheet Reporter application in the desktop on the left and the Options panel on the right. To select the Option panel, click **Setting** in the primary panel and select **Setting** from the pull-down menu. Here you can select the type of data that you want to be shown in the spreadsheet tables. These tables generate graphics depicting the sysout post processor RMF reports.

Z/OS V1R11 RMF SR

RMF Spreadsheet Reporter - Java Edition: Version 5 of the RMF Spreadsheet Reporter is a major revision with z/OS V1R11. The task-oriented concept of the previous versions is replaced by a more intuitive, resource-oriented user interface. With the new Spreadsheet Reporter version, you can create graphical charts from raw SMF data in a single step. You can directly start the RMF-provided spreadsheet macros from the Spreadsheet Reporter main window.

You access RMF SR from the panel shown in Figure 3-2 on page 178. The panel provides a brief description of the features of the workstation package version.

3.14 RMF Performance Monitor

- ❑ Monitoring z/OS performance from a workstation
 - Prerequisites for the client
 - Windows 2000, Windows XP, or Windows Vista
 - Netscape Navigator 6.0 or higher
 - Microsoft Internet Explorer 6.0 or higher
 - Mozilla Firefox 2.0 or higher
- ❑ View available metrics for a system or sysplex resources using one of the following monitoring applications:
 - RMF PM
 - RMF Monitor III Data Portal for z/OS
- ❑ z/OS prerequisites
 - RMF Distributed Data Server active
 - UNIX System Services full function mode
 - TCP/IP

Figure 3-14 RMF PM Java Edition

RMF Performance Monitoring

RMF Performance Monitoring (RMF PM) allows you to monitor the performance of your z/OS host from a workstation through a TCP/IP interface to one or more z/OS sysplexes. You log on to any sysplex and you can monitor the resources in the corresponding sysplex.

Client workstations for Windows

For Windows 2000 or later, to install and use the RMF PM you must have Standard User level access (Windows XP: Computer Administrator account type) or higher (see the Windows user management facilities). If you relate to the Restricted User level access group (Windows XP: Limited account type), you will encounter problems during the installation and use of the RMF PM.

Note: To install the RMF PM on Windows Vista, you need administrator rights for this workstation.

z/OS prerequisites

RMF PM uses a server on the z/OS sysplex to retrieve its performance data. This server is called the RMF Distributed Data Server. This host component is installed automatically.

See 3.15, “RMF Distributed Data Server (DDS)” on page 199 for information about how to customize and to start the RMF Distributed Data Server.

Also ensure that the following prerequisites are met on your z/OS host:

- ▶ UNIX System Services must be configured in “full function mode”.
- ▶ TCP/IP under UNIX System Services must be configured and active.

RMF PM and DDS

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

If you want to monitor systems in a sysplex using RMF PM, you must set up a DDS host session on that system in the sysplex with the highest RMF release.

RMF PM provides a selected subset of the information provided by the RMF Monitor III gatherer, such as general performance data and performance data for jobs. For systems running in goal mode, it provides workload-related performance data such as:

- ▶ WLM workloads
- ▶ WLM service classes
- ▶ WLM service class periods
- ▶ WLM report classes

3.15 RMF Distributed Data Server (DDS)

- ❑ DDS provides the ability to serve multiple clients in a single-server address space
 - Used by RMF PM
- ❑ Start the Distributed Data Server automatically
 - Using the DDS option
 - Ensures one active instance of the DDS in a sysplex
 - One system becomes the master system
- ❑ Following prerequisites are met on your z/OS host:
 - UNIX System Services must be configured in "full function mode"
 - TCP/IP under UNIX System Services must be configured and active
- ❑ GPMSRVxx parmlib member is needed

Figure 3-15 RMF Distributed Data Server (DDS)

RMF Distributed Data Server (DDS)

The Distributed Data Server (DDS) provides the ability to serve multiple clients in a single-server address space. This capability will be used, for example, by RMF PM.

You can start the Distributed Data Server automatically by using the DDS option. This option ensures that you always have one active instance of the Distributed Data Server within your sysplex. As soon as the RMF Sysplex Data Server recognizes the DDS option on any system in the sysplex, the Distributed Data Server is started on the RMF master system. The master system is the system with an active Monitor III gatherer and the highest z/OS release. If another system becomes the master system, the Distributed Data Server is automatically restarted on this system.

Starting the DDS option

You can specify the DDS option as follows:

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

You can stop the sysplex-wide DDS management using the NODDS option. For example, the command `F RMF,NODDS` also shuts down the current sysplex-wide DDS instance, regardless of on which system the command has been entered.

Using the DDS

On the systems that you want to monitor, you must start the Monitor III gatherer with identical MINTIME and SYNC options.

Also ensure that the following prerequisites are met on your z/OS host:

- ▶ UNIX System Services must be configured in “full function mode.”
- ▶ TCP/IP under Unix System Services must be configured and active.

GPMSRVxx parmlib member

The preparation of a Distributed Data Server (DDS) host session as a server address space for users of RMF PM or other exploiters requires the customization of a GPMSRVxx Parmlib member that is needed for the GPMSEVE procedure to start the Distributed Data Server.

3.16 RMF Monitor III Data Portal for z/OS

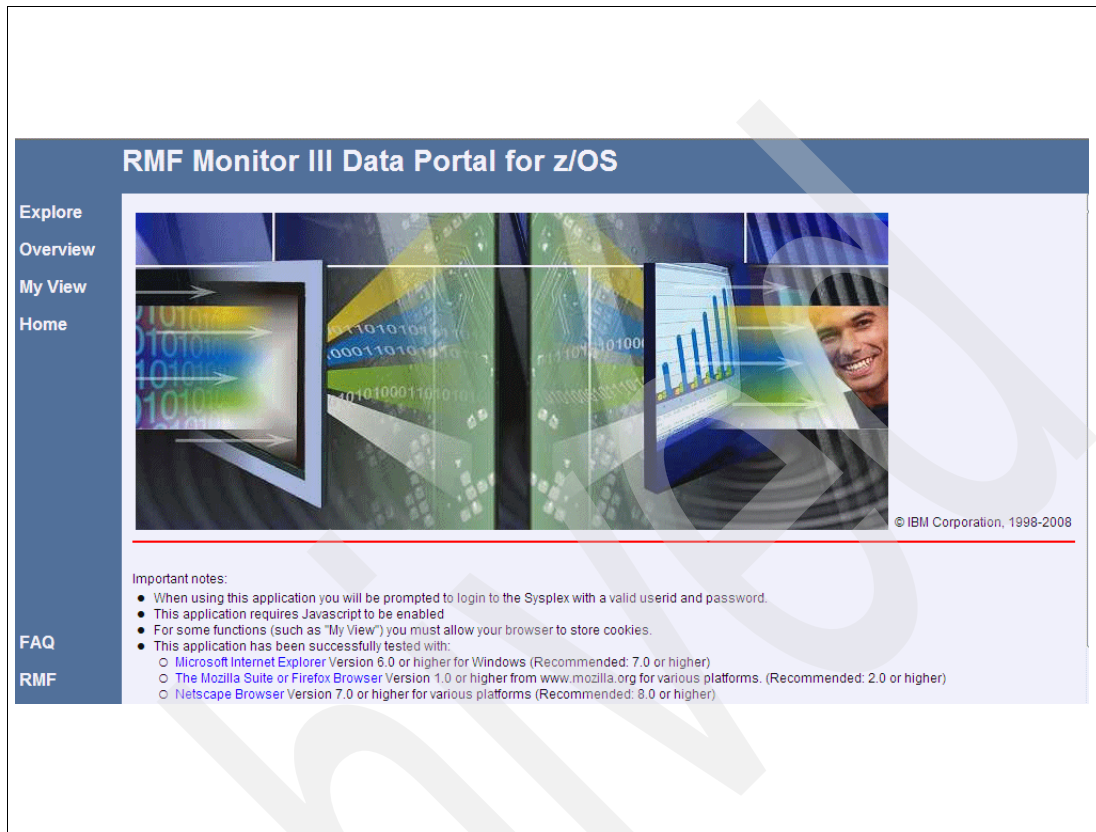


Figure 3-16 RMF Monitor III Data Portal

RMF Monitor III Data Portal

Instead of using the RMF PM, you can access RMF Monitor III data using DDS by using any web browser. The RMF Distributed Data Server (DDS) has been enhanced to respond directly to HTTP requests so that DDS behaves like an HTTP server. You can then open a web browser and simply type the following URL by specifying the host name of a system in your sysplex and the port number 8803, as follows:

```
http://<hostname>:8803  
For example:  
http://wtsc65.itso.ibm.com:8803.
```

Note: This URL requires that you know what system the DDS is executing on.

```
http://wtscplx2dds.itso.ibm.com:8803
```

Note: In this URL, wtscplx2 is the NJE node name for the sysplex.

Using this URL, you do not need to know the system where DDS is executing. Also, you can have multiple DDS sessions active at the same time.

Creating an automatic DDS connection

Starting up the RMF control session is enhanced to manage the DDS address space GPMERVE across the sysplex. When RMF initialization is complete, you can start the DDS

automatically on the best-suited system of the sysplex. To be able to access the DDS and not know what system GPMSEVE is executing on, implement the following:

- ▶ In each system's TCP/IP profile, define the following definitions because this makes the address 9.12.5.25 available as a dynamic IP address:

```
VIPADYNAMIC
VIPARANGE DEFINE 255.255.255.255 9.12.5.25 ; GPMSEVE
```

GPMSEVE procedure

Figure 3-17 displays a copy of the GPMSEVE procedure. Note that the MODDVIPA program is a part of TCP/IP. The sequence proceeds as follows:

- ▶ First, DELDVP deletes the dynamic address in case it was left on another system in the sysplex where GPMSEVE last ran.
- ▶ Then ADDDVP assigns the address 9.12.4.25 to the current system where GPMSEVE starts.
- ▶ Next, STEP1 is GPMSEVE itself.
- ▶ Finally, DELDVP deletes the dynamic address when GPMSEVE stops.

You can use the MODDVIPA utility to activate or delete a dynamic VIPA. The utility can be initiated from JCL or an OMVS script. MODDVIPA must be loaded from an APF authorized library and be executed under a user ID with superuser authority. The user ID must also have an OMVS segment defined (or defaulted). If the new profile for the MODDVIPA program has been defined under RACF, any user ID can execute the MODDVIPA program simply by being permitted to use this profile.

```
/* delete DVIPA if it was hanging around
//DELDVP EXEC PGM=MODDVIPA,REGION=OM,TIME=NOLIMIT,
// PARM='POSIX(ON) ALL31(ON)/-p TCPIP -d 9.12.5.25'
/* add DVIPA
//ADDVDP EXEC PGM=MODDVIPA,REGION=OM,TIME=NOLIMIT,
// PARM='POSIX(ON) ALL31(ON)/-p TCPIP -c 9.12.5.25'
//STEP1 EXEC PGM=GPMDDSRV,REGION=OM,TIME=NOLIMIT,
// PARM='TRAP(ON)&MEMBER'
/* PARM='TRAP(ON),ENVAR(ICLUI_TRACETO=STDERR)&MEMBER'
/*STEPLIB DD DISP=SHR,DSN=CEE.SCEERUN
/* DD DISP=SHR,DSN=CBC.SCLBDLL
//GPMINI DD DISP=SHR,DSN=SYS1.SERBPWSV(GPMINI)
//GPMHTC DD DISP=SHR,DSN=SYS1.SERBPWSV(GPMHTC)
//CEEDUMP DD DUMMY
//SYSPRINT DD DUMMY
//SYSOUT DD DUMMY
/* delete DVIPA
//DELDVP EXEC PGM=MODDVIPA,REGION=OM,TIME=NOLIMIT,
// PARM='POSIX(ON) ALL31(ON)/-p TCPIP -d 9.12.5.25'
```

Figure 3-17 GPMSEVE procedure

3.17 RMF Monitor III Data Portal for z/OS Overview panel

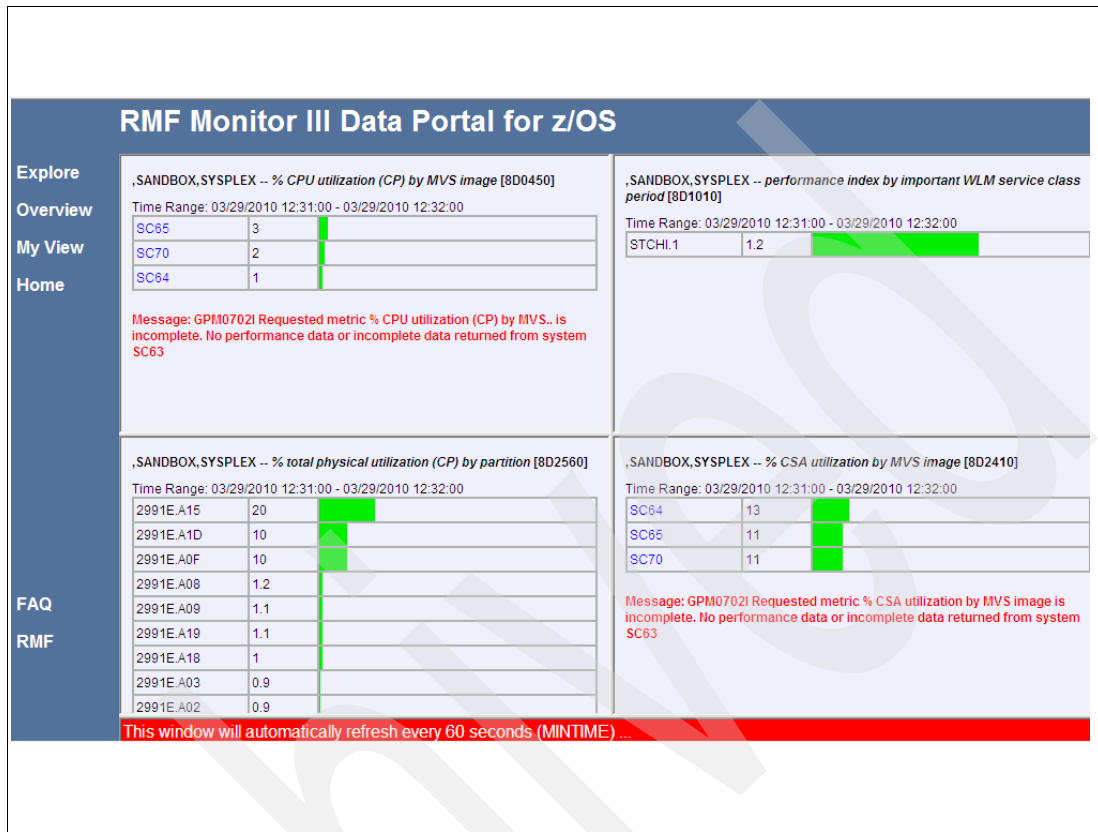


Figure 3-18 RMF Data Portal Overview panel

HTTP server for port 8803

DDS behaves as an HTTP server for port 8803. After you customize the first window that appears, you see the panel shown in Figure 3-16 on page 201. The left side of the panel offers the following options:

Overview	This presents a health check overview of your system.
My View	This shows views where you can see your preferred metrics.
Explore	This allows you to explore the resources and performance metrics of your system.
RMF	This leads you to the RMF Home page.
Home	This leads you back to the welcome window.

Overview panel

When you click **Overview** in Figure 3-16 on page 201, you reach the Overview panel shown in Figure 3-18. As shown in the figure, you see the following views:

- ▶ Average logical PU utilization per logical partition
- ▶ Performance index for the most important device classes
- ▶ Average physical PU utilization per logical partition
- ▶ CSA virtual storage utilization per MVS logical partition

Note: The message This window will automatically refresh every 60 seconds (MINTIME) ... is displayed at the bottom of this panel.

3.18 RMF Monitor III Data Portal for z/OS Explore panel

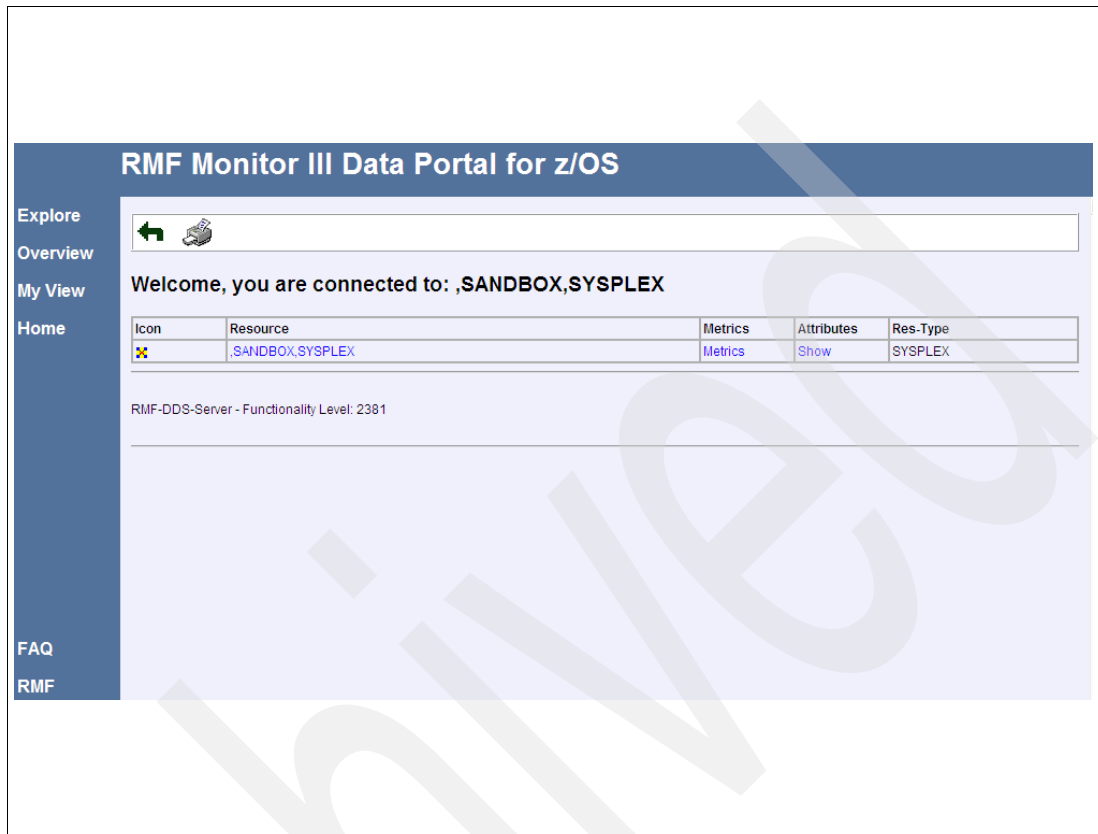


Figure 3-19 Data Portal Explore panel

Data Portal Explore panel

By selecting **Explore**, you reach the panel shown in Figure 3-19. The panel offers the following options:

- ▶ Resource (click your sysplex name; in Figure 3-19, this is **SANDBOX,SYSPLEX**)
- ▶ Metrics (click **Metrics**)
- ▶ Attributes (click **Show**)

By selecting the Metrics button in the Explore panel you can choose a full RMF Monitor III report type such as CACHDET or SYSSUM, or available metrics, as shown in 3.19, “RMF Monitor III Data Portal for z/OS Metrics Explore panel” on page 205.

3.19 RMF Monitor III Data Portal for z/OS Metrics Explore panel

RMF Monitor III Data Portal for z/OS

Explore
Overview
My View
Home
FAQ
RMF

Full RMF Reports:

CACHDET	CACHSUM	CFACT	CFOVER	CFSYS	SPACED	SPACEG	SYSSUM	XCFOVW	XCFFPATH	XCFSYS
---------	---------	-------	--------	-------	--------	--------	--------	--------	----------	--------

Available metrics for: ,SANDBOX,SYSPLEX

Metric description	Help	Id
% delay	Explanation	8D0160
% delay for enqueue	Explanation	8D1A20
% delay for i/o	Explanation	8D1A80
% delay for operator	Explanation	8D1AE0
% delay for processor	Explanation	8D1B40
% delay for storage	Explanation	8D1BA0
% delay for swsub	Explanation	8D1C00
% using	Explanation	8D04A0
% using for i/o	Explanation	8D1D40
% using for processor	Explanation	8D1DB0
% workflow	Explanation	8D0550
% workflow for i/o	Explanation	8D1ED0

Figure 3-20 RMF Data Portal report options and metrics panel

RMF Data Portal report options and metrics panel

From this panel, when you select a full RMF report type such as CACHDET and CFSYS, you reach a panel that is similar to the host Monitor III panel.

Full RMF reports

Here you can perform actions like those you perform in a host terminal, such as going backward in interval time periods as much as your SMF buffer limits. This is shown in 3.20, “RMF Portal CACHDET report panel” on page 207 and 3.21, “RMF Monitor III Data Portal for z/OS My View panel” on page 209.

CACHDET The CACHDET report provides detailed information about the activities of one cache subsystem.

Note: An example of this report is shown in Figure 3-21 on page 207.

CACHSUM The Cache Summary report (CACHSUM) provides an overview of the activities in the cache subsystem for all SSIDs. You can use this as starting point, when you analyze I/O performance, to obtain a first impression of the I/O processing.

CFACT The Coupling Facility Activity report (CFACT) provides you with information about the activities in each structure. You can use this report for detailed

analysis of each structure in your Coupling Facilities. You see the type of a structure and the activities from each system that uses a structure.

- CFOVER** The Coupling Facility Overview report (CFOVER) provides you with information about all Coupling Facilities that are connected to the sysplex. You can start investigating the performance of the Coupling Facilities in your sysplex by using the CFOVER report.
- CFSYS** The Coupling Facility Systems report (CFSYS) provides you with information about the distribution of Coupling Facility requests among the systems and about the activities in the subchannels and paths attached to the Coupling Facilities in the sysplex.
- SPACED** The Disk Space Report displays capacity and disk space information for volumes. This report displays only those volumes that belong to storage groups specified with the Monitor III SGSPACE gatherer option. You can use this information to decide whether a certain volume provides sufficient free disk space for new allocation requests.
- SPACEG** A storage group is a collection of storage volumes and attributes, defined by the storage administrator and treated as a single object storage hierarchy. The Storage Space Report allows you to keep track of disk space consumption on a storage group level. This report displays only volumes that belong to storage groups specified with the Monitor III SGSPACE gatherer option.
- SYSSUM** The Sysplex Summary (SYSSUM) report allows service administrators and performance analysts to see at a glance whether service goals are being satisfied.
- XCFGROUP** This report provides XCF group statistics for every defined XCF group.
- XCFOVW** The XCF systems overview report provides system information about every LPAR in the sysplex.
- XCFPATH** The Channel Path Activity report (CHANNEL) provides you with information about channel path activity for all channel paths in the system. The report contains data for every channel path that is online during data gathering.
- XCFSYS** This report provides XCF sysplex statistics for every transport class for every connection in the sysplex.

Available metrics

When working with the numerous metrics (note that only a small number of them are displayed in Figure 3-20 on page 205) with the Data Portal, you need to understand the concept underlying RMF Monitor III resources and the associated metrics.

Each metric has a Help explanation describing what the metric displays. For example, for % delay, which is the first metric displayed in Figure 3-20 on page 205, using Help displays text explaining what this metric is about.

You can save examples of metrics by selecting, on the panel, to add that metric to “My View”. Selected “My Views” are shown in Figure 3-23 on page 209.

3.20 RMF Portal CACHEDET report panel

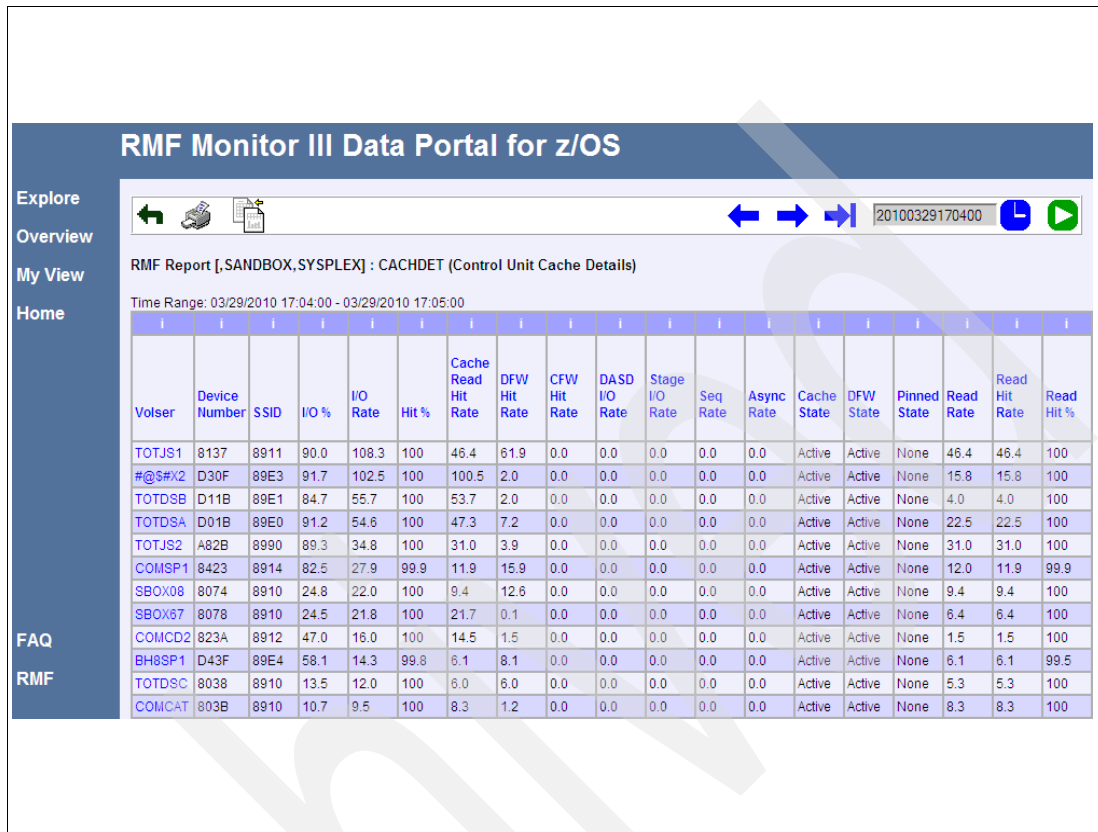


Figure 3-21 Data Portal - CACHEDET full RMF report panel

Data Portal - CACHEDET RMF report panel

This section examines the CACHEDET report, shown in Figure 3-21, as an example of how to use these reports. If you place the cursor on any field of the first two columns, a pop-up panel appears that displays details of the selected volume.

Cursor-sensitive control of the third column leads you to a pop-up panel showing details of that SSID. If you select VOLSER TOTJS1, the pop-up panel appears, displaying the information shown in Figure 3-22 on page 208.

Note: In Figure 3-22 on page 208, not all data is displayed.

RMF Report - One Row [,SANDBOX,SYSPLEX] : CACHDET (Control Unit Cache Details)

Time Range: 03/29/2010 17:14:00 - 03/29/2010 17:15:00

i	Volser	TOTJS1
i	Device Number	8137
i	SSID	8911
i	I/O %	89.9
i	I/O Rate	108.1
i	Hit %	100
i	Cache Read Hit Rate	46.3
i	DFW Hit Rate	61.8
i	CFW Hit Rate	0.0
i	DASD I/O Rate	0.0

Figure 3-22 CACHDET report displaying information about the first VOLSER shown

Additional information

In Figure 3-21 on page 207, the columns display the following information:

- ▶ There are several 3390 devices sorted by I/O rate.
- ▶ The columns to the right of the I/O rate contain information about DASD cache performance.
- ▶ Due to the current technology and algorithms, the most important figure is Read Hit%. Notice that all devices in this page panel are at the 100% mark.
- ▶ Scrolling down you see smaller numbers. These figures are to be, by a rule of thumb, greater than 80%. When less than 80%, chances are that you are dealing with a DASD cache-unfriendly application such as DB2. In this case, the best choice is to map such volumes to fast real disks.

3.21 RMF Monitor III Data Portal for z/OS My View panel

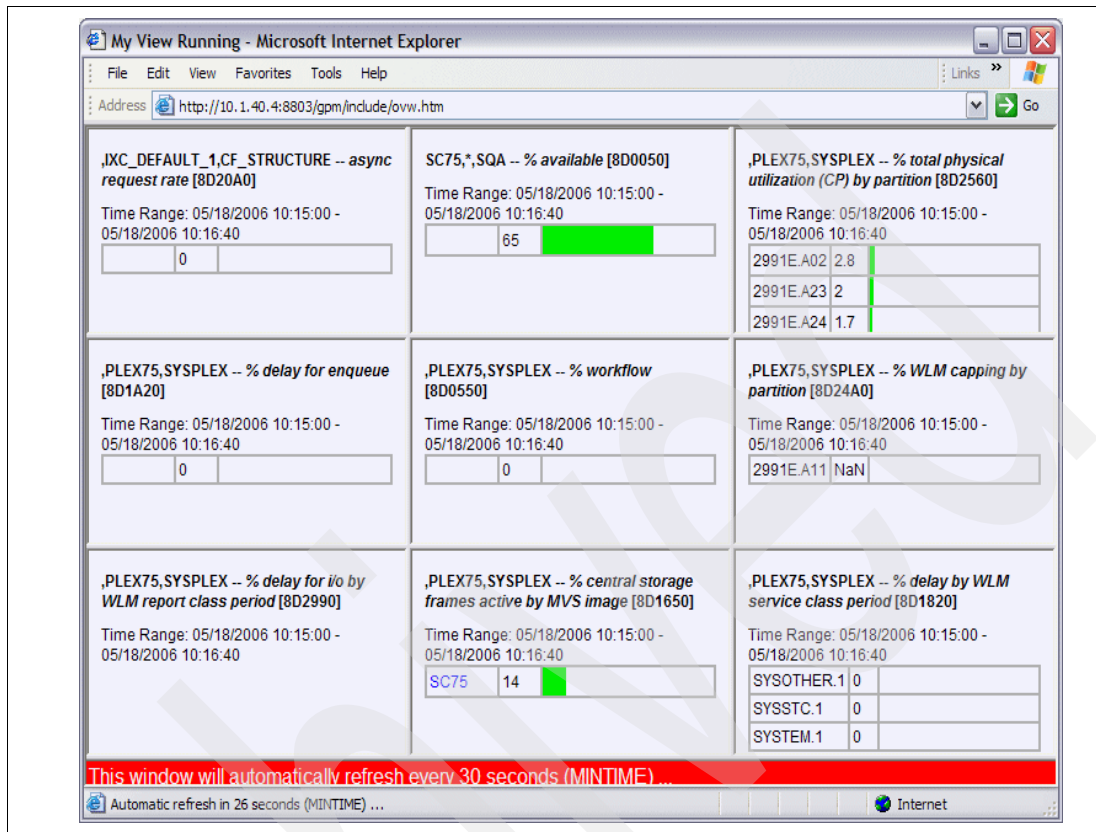


Figure 3-23 Data Portal - My View display

Data Portal - My View display

The Distributed Data Portal allows you to choose your own report by selecting which metrics you want. About 80% of all data fields in Monitor III can be chosen as metrics. In the future, all fields in Monitor III can be chosen as metrics.

Add metric to My View

When viewing a metric, you can add it to My View by selecting the Add to My View button that is at the bottom of the metric display. Any of the panels of the RMF Data Portal that display on the leftmost side, like the panel shown in Figure 3-22 on page 208, allow you select My View. Figure 3-23 shows an example of all views that were added to My View.

Any of the panels of the RMF Data Portal that display on the leftmost side, like the panel shown in Figure 3-22 on page 208, allow you select My View. My View then displays the views you selected; see Figure 3-23.

3.22 RMF enhancements for z/OS V1R11

- ❑ New report on OPT parameter settings
- ❑ Replacement of the Postprocessor Plot report
- ❑ New real storage measurements
- ❑ Measuring WLM's promotion for workloads holding locks
- ❑ RMF Postprocessor reports in XML format
- ❑ New overview conditions for special purpose processors
- ❑ New overview conditions for HiperDispatch
- ❑ Selective DDS deactivation of Monitor III reports
- ❑ New installation process for RMF workstation components



Figure 3-24 RMF enhancements for z/OS V1R11

RMF enhancements

This section describes the RMF enhancements for z/OS V1R11.

New report for OPT

RMF provides a new Monitor II OPT Settings report that displays information about the active OPT PARMLIB member and the settings of all OPT parameters.

Replacement of the Postprocessor Plot report

Plot reports by the RMF Postprocessor are no longer available. Each type of plot report can be replaced by a corresponding Overview control statement. After creating Overview records, the Spreadsheet Reporter with the Open RMF Overview Spreadsheets macro can be used for graphical presentation of Postprocessor data.

New real storage measurements

The Frame and Slot Counts section of the Postprocessor Paging Activity report is enhanced to provide measurements about real storage requests and page faults.

Measuring WLM's promotion for workloads holding locks

The Postprocessor Workload Activity report is enhanced to show the CPU time consumed for work units while they have been promoted by WLM to free a local suspend lock faster. In

addition, RMF provides new overview conditions for the Postprocessor based on SMF record 72-3.

RMF Postprocessor reports in XML format

The Postprocessor can now generate certain reports in XML format. By specifying new ddnames for the Postprocessor output, reports in XML format instead of text reports are generated. Currently available in XML format are the CPU, CRYPTO, ESS, FCD, and OMVS reports and the Overview report. You can use the RMF Spreadsheet Reporter to request the reports in XML format for display in a web browser.

New overview conditions for special purpose processors

RMF provides new overview conditions for the Postprocessor based on SMF record 70 for measuring the CPU activity of zAAPs and zIIPs.

New overview conditions for HiperDispatch

RMF provides new overview conditions for the Postprocessor based on SMF record 70 for systems running in HiperDispatch mode.

Selective DDS deactivation of Monitor III reports

You can temporarily deactivate selected Monitor III reports that are not required by any monitoring client from being provided by the Distributed Data Server.

New installation process for RMF workstation components

The installation process for the RMF Performance Monitoring (RMF PM) and the RMF Spreadsheet Reporter has changed.

3.23 Cryptographic hardware information

- ❑ RACF support for cryptographic hardware
- ❑ RACF Crypto Hardware Activity report
 - Cryptographic coprocessors
 - Cryptographic accelerators
 - ICSF Services
- ❑ Data is important for performance problem analysis and capacity planning
- ❑ Additional card can be plugged
 - Express3 Coprocessor crypto card



Figure 3-25 Cryptographic hardware information

RACF cryptographic hardware support

With the availability of the cryptographic hardware (PCI Cryptographic Coprocessors and PCI Cryptographic Accelerators) on an LPAR basis, RMF provides performance monitoring with the new Postprocessor Crypto Hardware Activity report, which is based on new SMF records type 70 subtype 2. These records will be gathered by the new Monitor I gathering option CRYPTO. In addition, new overview conditions are available for the Postprocessor.

RACF Crypto Hardware Activity report

The Crypto Hardware Activity report provides information about the activities in the various cryptographic hardware functions. Most cryptographic hardware functions can only be used through Cryptographic Support for z/OS (ICSF). ICSF is a standard component of z/OS. It provides cryptographic services in the z/OS environment. The report provides the following sections:

▶ Cryptographic coprocessors

This feature provides secure cryptographic functions, use of secure encrypted key values, clear key and secure PKA operations, and special user cryptographic functions (using the user defined extension (UDX) capability of the card).

For cryptographic coprocessors, give special attention to RSA key-generation operations because these operations require a high amount of cryptographic processing capacity. Therefore, they are reported in addition to the total number of operations.

- ▶ **Cryptographic accelerators**

This feature provides public key operations such as those used with Secure Sockets Layer (SSL) or Transport Layer Security (TLS) protocols, which are widely used to help secure e-business applications. The data for cryptographic accelerators shows details for the two available algorithms, modular exponentiation (ME) and Chinese Remainder Theorem (CRT) for both key lengths (1024 and 2048 bit). This provides information about how the use of these algorithms affects accelerator utilization.

- ▶ **ICSF Services**

The Crypto Hardware Activity report provides performance measurements on selected ICSF activities.

RMF report for capacity planning

The Enterprise Disk Systems report provides measurements about the activities of an enterprise disk system. RMF monitors the activity on an enterprise disk system independently from the source of the activity. Activity can be caused by the z/OS system on which RMF is running, or from any other system using the enterprise disk system. You can use the data contained in this report for checking your current disk configuration, for bottleneck analysis, and for capacity planning.

The Internet is becoming the basis for electronic commerce. More businesses are automating their data processing operations. Many businesses transmit sensitive data on open communication networks and store confidential data offline. Every day the potential for unauthorized persons to access sensitive data increases. Only cryptographic services can provide the data confidentiality and the identity authentication that is required to protect business commerce on the Internet.

ICSF software element

ICSF is a software element of z/OS that works with the cryptographic hardware and the z/OS Security Server to provide secure, high-speed cryptographic services. ICSF provides APIs by which applications request the cryptographic services. The cryptographic feature available to the applications depends on the server or processor hardware.

3.24 Cryptographic coprocessors

- ❑ RMF now provides performance monitoring with the new Postprocessor Crypto Hardware Activity report
 - SMF Record 70, Subtype 2

CRYPTO HARDWARE ACTIVITY												PAGE 6		
z/OS V1R11				SYSTEM ID SYS1		DATE 11/28/2009		INTERVAL 14.59.946						
----- CRYPTOGRAPHIC COPROCESSOR -----				RPT VERSION V1R11 RMF		TIME 16.30.00		CYCLE 1.000 SECONDS						
TYPE	ID	RATE	EXEC	TIME	UTIL%	KEY-GEN								
PCIXCC	0	0.00	0.0	0.0	0.0	RATE								
	1	0.01	3205	32.1	0.01									
	2	83.04	1.1	8.8	0									
	3	0.00	0.0	0.0	0.00									
CEX2C	4	210.8	4.4	93.3	1.91									
	5	186.4	4.8	89.6	1.85									
----- CRYPTOGRAPHIC ACCELERATOR -----														
TYPE	ID	RATE	EXEC	TIME	UTIL%	ME(1024)		ME(2048)		CRT(1024)		CRT(2048)		
PCICA	6	165.2	1.3	21.5	107.1	1.1	11.8	0.00	0.0	0.0	58.1	1.7	9.7	0.00
	7	892.3	3.6	64.3	350.1	4.1	28.6	0.00	0.0	0.0	512.6	2.4	24.7	29.65
	8	684.8	3.5	47.8	260.4	4.0	21.0	0.00	0.0	0.0	402.4	2.3	18.6	22.02
----- ICSF SERVICES -----														
DES ENCRYPTION		DES DECRYPTION		MAC		SHA-1		HASH		SHA-512		PIN		
SINGLE TRIPLE		SINGLE TRIPLE		GENERATE VERIFY		SHA-1 SHA-256		SHA-123K		TRANSLATE		VERIFY		
RATE 4975K 497.5		12438 1244K		12438 4975K		497.5 0.00		123K		1244K		1244K		
SIZE 0.75 100K		10.00 0.01		10.00 0.01		10000 0.00		348.0						

Figure 3-26 Cryptographic coprocessors

Crypto Hardware Activity report

The data shown for cryptographic coprocessors and accelerators always reflects the total activity in your CPC. The data shown for ICSF services is for the partition. If measurement data for one of the cryptographic features is not available, the corresponding report section is omitted.

The Crypto Hardware Activity report provides performance measurements on selected ICSF activities, as explained here.

- ▶ Using the Data Encryption Standard (DES). This is probably the best known and scrutinized encryption algorithm, to encipher and decipher data.
- ▶ Generating and verifying message authentication codes (MAC). The MAC is a value calculated from the message according to a secret shared DES key and sent to the receiver together with the message. The receiver can recalculate the MAC and compare it with the MAC received. If the MAC values are identical, it means that the message has not been altered during transmission.
- ▶ Using public hash functions. A hash is calculated from the transmission data according to a public key or function in cases where it is impossible to share a secret key. If the recalculated hash is identical to the one calculated before transmission, data integrity is ensured.
- ▶ Translating and verifying PINs.

Report data

Type that defines the cryptographic coprocessor:

PCICC	PCI Cryptographic Coprocessor
PCIXCC	PCI-X Cryptographic Coprocessor
CEX2C	Crypto Express2 Coprocessor
ID	Index that specifies the cryptographic coprocessor
RATE	Rate of all operations on this cryptographic coprocessor.
EXEC TIME	Average execution time (milliseconds) of all operations on this cryptographic coprocessor
UTIL%	Total utilization percentage of this coprocessor

SMF record 70 subtype 2

With the availability of the new cryptographic hardware (PCI Cryptographic Coprocessors and PCI Cryptographic Accelerators) on an LPAR basis, RMF now provides performance monitoring with the new Postprocessor Crypto Hardware Activity report, which is based on new SMF records type 70 subtype 2.

3.25 RMF Workload Activity - Crypto (CRY)

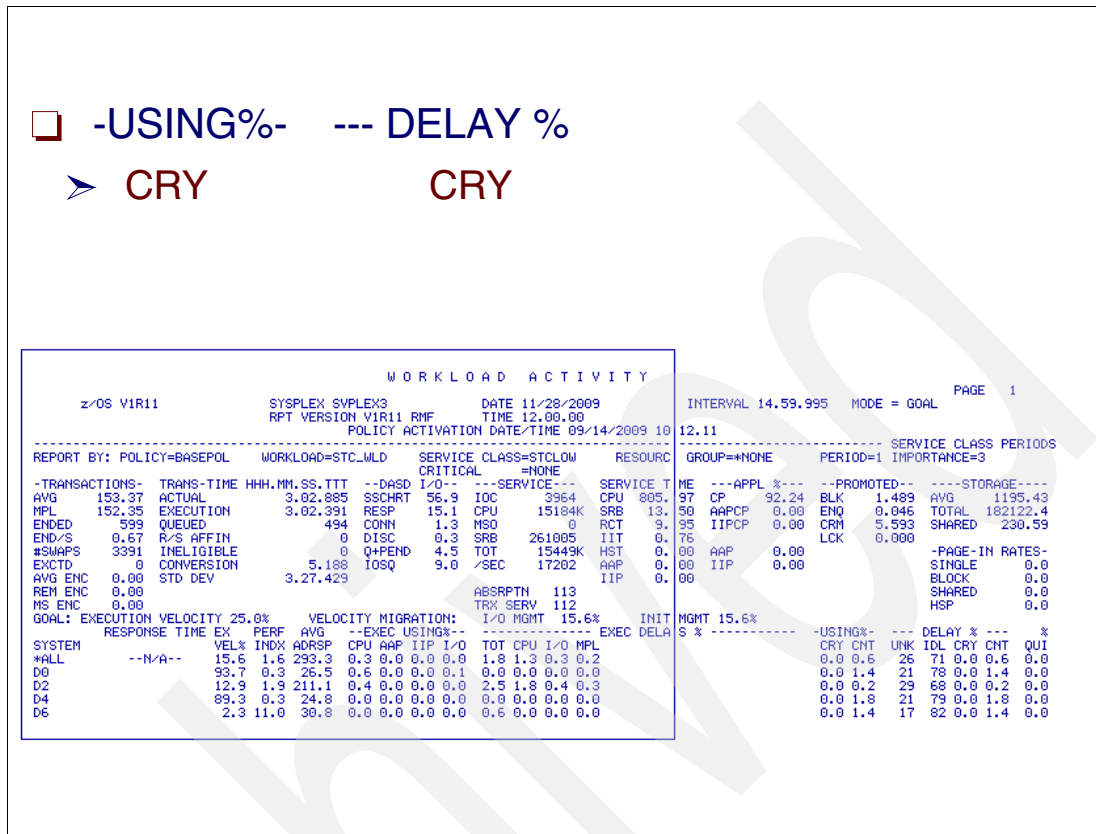


Figure 3-27 RMF Workload Activity report showing crypto information

RMF Workload Activity report

Figure 3-27 shows a report for service class STCLOW where the first service class period (PERIOD=1) is defined with an execution velocity goal. For example, to produce the report for all service class periods of service class STCLOW, specify:

```
SYSRPTS(WLMGL(SCPER(STCLOW)))
```

Crypto information in the report (z/OS V1R11)

The fields specifying the crypto information are listed here.

USING% Percentage of using states:

CRY - Crypto using state. A TCB or SRB was found to be using a cryptographic asynchronous message processor (CAP) or an adjunct processor (AP).

CNT - Contention using state. Work is holding resources.

Delay % CRY - Crypto delay state. A TCB or SRB was found to be waiting for a CAP, an AP, or a processor feature queue.

CNT - Contention delay state. Work is waiting for resources.

3.26 RMF Coupling Facility reports

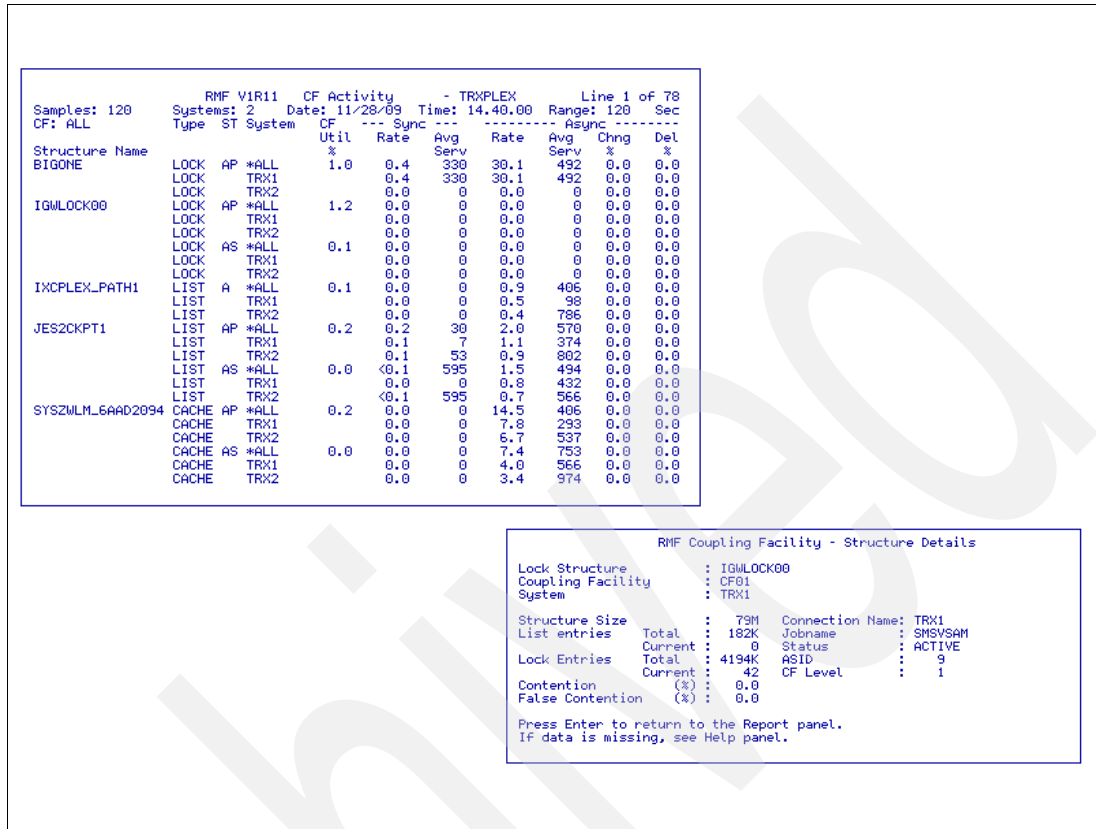


Figure 3-28 Coupling Facility reports

System-managed Coupling Facility (CF) duplexing

Coupling Facility (CF) duplexing ensures high application availability in a Parallel Sysplex environment. The performance management aspects that have to be covered for CF duplexing are provided by a Coupling Facility Activity report about new peer CF connectivity. This enables you to evaluate and monitor your CF configuration, and you can apply the necessary changes to tune accommodation to new structure instances resulting from system-managed duplexing.

Coupling Facility Activity (CFACT) report

The Coupling Facility Activity report (CFACT) gives you information about the activities in each structure. You can use this report to analyze in detail each structure in your Coupling Facilities. You see the type of a structure and the activities from each system that uses a structure. If you want to obtain further details, you will receive them through cursor-sensitive control. A pop-up panel shows the allocation details and the name of the address space that is currently using the structure. If you experience performance problems for one or several structures in your Coupling Facilities, investigate the appropriate applications.

Monitor III CFACT - Coupling Facility Activity report

The CFACT report is extended with report column CF Util%. This column displays the CF processor utilization of an active structure. This value is only available with a CF level greater than 14 and the data has been collected with the CFDETAIL option. Otherwise, N/A is displayed.

Structure status (ST) column

The structure status displayed in column ST is enhanced, as follows:

- N** New. The structure became allocated and connected during the report interval.
- A** Active. The structure is allocated and connected.
- AP** Active Primary. The structure is the rebuild-old in the duplexing rebuild process.
- AS** Active Secondary. The structure is the rebuild-new in the duplexing rebuild process.

Note: Data gathering for this report is enabled by default, using the gathering option CFDETAIL. With CFDETAIL, a large amount of data is being gathered that enables you to obtain many details about the usage of each structure in the Coupling Facility.

Keep in mind that this data gathering is done only on one member of the sysplex. It is known as *sysplex master gathering*, and it has been implemented to reduce performance overhead on non-master members and to reduce the amount of data in SSHs and SMF records. The RMF Sysplex Data Server determines internally which member of the sysplex will be the master. This cannot be controlled externally by the operator or system administrator and is transparent for performance monitoring.

3.27 CF-to-CF activity reporting

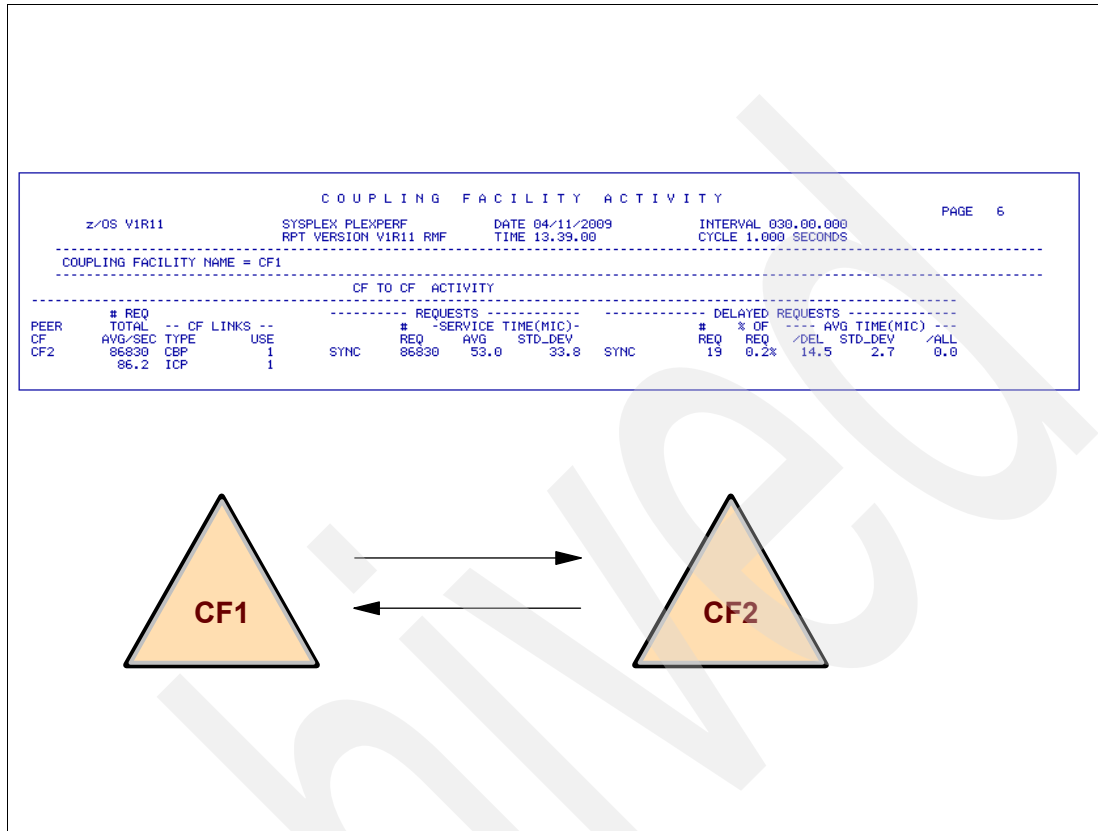


Figure 3-29 CF-to-CF activity reporting

CF-to-CF Activity section

System-managed duplexing requires links between the Coupling Facilities to synchronize the structure updates. The CF-to-CF Activity section of the Coupling Facility Activity report provides the information to monitor CF-to-CF link performance and utilization. Figure 3-29 shows an example of the CF-to-CF Activity section.

CF report information

The report shown in Figure 3-29 displays field details.

PEER CF This is the name of the remote Coupling Facility.

REQ TOTAL This is the total number of requests delayed in the interval. This is the sum of the following signals which have been sent from the subject CF to this remote CF:

- ▶ Number of halt execution signals
- ▶ Number of ready to complete signals
- ▶ Number of ready to execute signals
- ▶ Number of request for suppression signals
- ▶ Number of request for suppression accepted signals

#REQ AVG/SEC This is the average number of requests per second.

3.28 HiperSockets: Client/Server in a mainframe machine

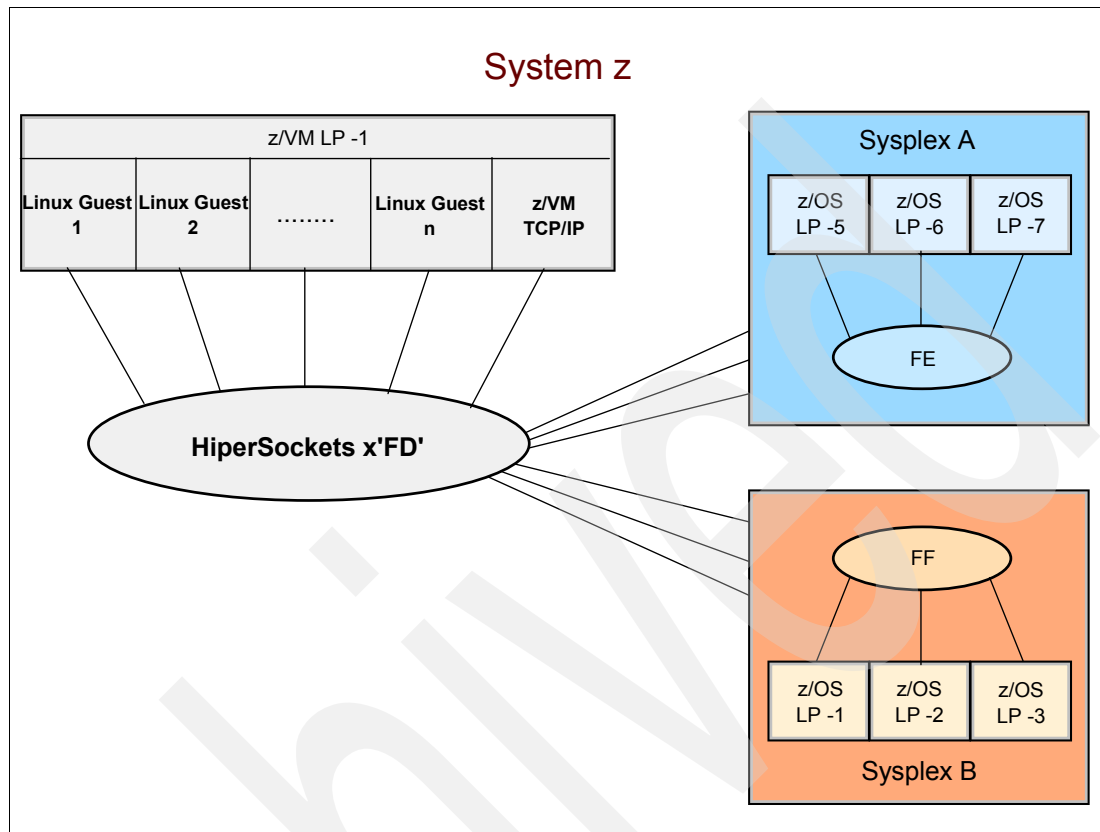


Figure 3-30 HiperSockets: Client/Server in a mainframe machine

HiperSockets: Client/Server in a mainframe machine

The three HiperSockets illustrated in Figure 3-30 are used as explained here.

- ▶ HiperSockets with CHPID FD
- ▶ Connected to this HiperSockets are all servers in the System z, which are:
 - The multiple Linux servers running under z/VM in LPAR-1
 - The z/VM TCP/IP stack running in LPAR-1
 - All z/OS servers in sysplex A (logical partitions 5 to 7) for non-sysplex traffic
 - All z/OS servers in sysplex B (logical partitions 8 to 10) for non-sysplex traffic

- ▶ HiperSockets with CHPID FE

This is the connection used by sysplex A (logical partitions 5 to 7) to transport TCP/IP user-data traffic among the three sysplex logical partitions.

If the following prerequisites are met, HiperSockets is automatically used within a single sysplex environment:

- XCF dynamics are defined to the TCP/IP stacks.
- HiperSockets is available to the TCP/IP stacks.

- ▶ HiperSockets with CHPID FF

This is the connection used by sysplex B (logical partitions 8 to 10) to transport TCP/IP data traffic among the three sysplex logical partitions.

HiperSockets

HiperSockets is a totally integrated, no charge, any-to-any virtual TCP/IP network that provides a list of benefits not achievable by grouping servers around a mainframe interconnected by external networking technology.

HiperSockets on zSeries hardware

zSeries HiperSockets is a microcode function of a zSeries machine. It is supported by the following operating systems:

- ▶ z/OS V1R2 and later
- ▶ z/OS.e
- ▶ z/VM V4R2 and later
- ▶ Linux for zSeries (64-bit mode) and Linux for S/390 (31-bit mode)

HiperSockets supports up to four independent virtual LANs, which operate as TCP/IP networks within a zSeries CEC. HiperSockets can be used to communicate among consolidated servers in a single processor. All the hardware machines running these separate servers can be eliminated, along with the cost, complexity, and maintenance of the networking components that interconnect them.

3.29 RMF enhancement to support HiperSockets

CHANNEL PATH ACTIVITY														PAGE 1		
z/OS VIR11		SYSTEM ID CB88		DATE 11/28/2009		INTERVAL 30.00.007										
IODF = 8E		CR-DATE: 09/14/2009		RPT VERSION VIR11 RMF		TIME 08.00.00		CYCLE 1.000 SECONDS								
		CR-TIME: 16.47.03		ACT: POR		MODE: LPAR		CPMF: EXTENDED MODE		CSSID: 1						
OVERVIEW FOR DCM-MANAGED CHANNELS																
CHANNEL GROUP	G NO	UTILIZATION(%)			READ(MB/SEC)		WRITE(MB/SEC)									
CNC-SM	4	0.82	1.13													
FC-SM	5 2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
FC-SM	4 2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	
DETAILS FOR ALL CHANNELS																
ID	TYPE	G	SHR	UTILIZATION(%)			READ(MB/SEC)		WRITE(MB/SEC)		FICON OPERATIONS			ZHPF OPERATIONS		
12	OSD	Y		0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00
14	OSD	Y		0.00	0.00	0.00	0.00	0.00	0.46	0.44	0.45					
16	OSD	Y		0.33	1.10		0.00	0.47	4.13	2.79	4.14					
20	CTC-S	Y		0.00	0.00											
27	CNC-S	Y		0.00	0.00											
2B	CNC-S	Y		1.23	4.90											
2C	CNC-S	Y		0.19	0.67											
30	FC-S	5	Y	0.24	32.72	8.49	0.05	49.57	0.00	0.24	208.6	2.8	0.6	0.0	0.0	0.0
31	FC-S	5	Y	0.22	32.27	8.21	0.05	47.79	0.00	0.24	192.6	1.6	0.0	0.0	0.0	0.0
36	FC-S	4	Y	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0	0.0	0.0
37	FC-S	4	Y	0.00	0.12	0.03	0.00	0.11	0.00	0.02	10.3	1.6	0.0	0.0	0.0	0.0
38	FC-S	4	Y	0.00	0.26	0.04	0.00	0.14	0.00	0.05	21.2	2.2	0.0	0.0	0.0	0.0
39	FC-S	4	Y	0.00	0.19	0.02	0.00	0.03	0.00	0.03	15.1	9.0	1.0	0.0	0.0	0.0
3A	FC-S	4	Y	0.00	0.17	0.02	0.00	0.03	0.00	0.03	14.6	2.6	0.1	0.0	0.0	0.0
3E	FC-S	4	Y	0.00	0.29	0.02	0.00	0.02	0.00	0.05	23.0	2.0	0.0	0.0	0.0	0.0
7C	CNC-SM	Y		3.08	4.27											
7D	CNC-SM	Y		0.08	0.09											
81	FC-S	3	Y	1.92	16.35	4.08	1.91	22.77	0.35	1.73	828.9	2.8	0.0	179.3	1.3	0.0
82	FC-S	5	Y	0.07	0.61	0.21	0.10	1.21	0.02	0.10	11.3	1.0	0.0	45.5	1.0	0.0
83	FC-S	5	Y	0.07	0.60	0.21	0.10	1.21	0.02	0.10	11.3	1.0	0.0	45.5	1.0	0.0
84	FC-S	4	Y	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.1	1.3	0.0	0.0	0.0	0.0
95	FC-S	3	Y	1.65	9.35	1.02	0.99	3.14	0.23	1.45	575.2	1.2	0.0	214.9	1.1	0.0
8C	FC-S	3	Y	1.71	11.67	1.54	0.70	6.37	0.10	1.05	803.6	1.9	0.0	0.0	0.0	0.0
A6	FC-SM	5	Y	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0	0.0	0.0
B6	FC-SM	5	Y	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0	0.0	0.0	0.0	0.0	0.0
CHANNEL	PATH	WRITE(B/SEC)		MESSAGE RATE		MESSAGE SIZE		SEND FAIL		RECEIVE FAIL						
ID	TYPE	G	SHR	PART	TOTAL	PART	TOTAL	PART	TOTAL	PART	TOTAL	PART	TOTAL			
E0	IQD	Y		0	60.85K	0	91	-----	669	0	0	0	0			
E1	IQD	Y		0	0	0	0	-----	-----	0	0	0	0			
E2	IQD	Y		0	0	0	0	-----	-----	0	0	0	0			
E3	IQD	Y		0	0	0	0	-----	-----	0	0	0	0			

Figure 3-31 HiperSockets monitoring

RMF support for HiperSockets monitoring

zSeries HiperSockets is a technology that provides high-speed TCP/IP connectivity between servers within a zSeries CEC. It eliminates the need for any physical cabling or external networking connection between these virtual servers. The communication is through the system memory of the processor. The virtual servers that are so connected form a "virtual LAN." HiperSockets uses internal Queued Input/Output (iQDIO) at memory speeds to pass traffic between the virtual servers.

RMF CPU Activity report

The following fields used are for HiperSockets:

- WRITE(B/SEC)** PART. This is the data transfer rates from the channel to the control unit for this partition.
 TOTAL. This is the data transfer rates from the channel to the control unit for the CPC.
 The values are shown in bytes/second.
- MESSAGE RATE** PART. This is the rate of messages sent by this partition.
 TOTAL. This is the rate of messages sent by the CPC.
- MESSAGE SIZE** PART. This is the average size of messages sent by this partition.
 TOTAL. This is the average size of messages sent by the CPC.

SEND FAIL PART Rate of messages (sent by this partition) that failed, but not caused by an unavailable buffer in the receiving partition.

RECEIVE FAIL PART. This is the rate of messages (received by this partition) that failed due to unavailable buffers. The value can indicate that more receive buffers are required.

TOTAL. This is the rate of messages (received by the CPC) that failed due to unavailable buffers.

Monitor III Utility fields

You can use the Monitor III Utility to customize the CHANNEL report in a way that the following additional values are shown. RMF builds ERBCHAT3 when using CHANNEL as a report type. The following fields are located in the tabular report data table ERBCHAT3.

The following fields are only available for HiperSockets:

CHACTMVC This is the total message sent rate.

CHACTSVC This is the total message sent size.

CHACTFVC This is the total receive fail rate.

CHACPMVC This is the rate of messages sent by this LPAR.

CHACPSVC This is the average size of messages sent by this LPAR.

CHACPFVC This is the rate of messages received by this partition that failed due to an unavailable buffer. The value can indicate that more receive buffers are required.

CHACSFVC This is the rate of messages sent by this partition that failed due to any other reason than unavailable buffers.

3.30 System Management Facility (SMF)

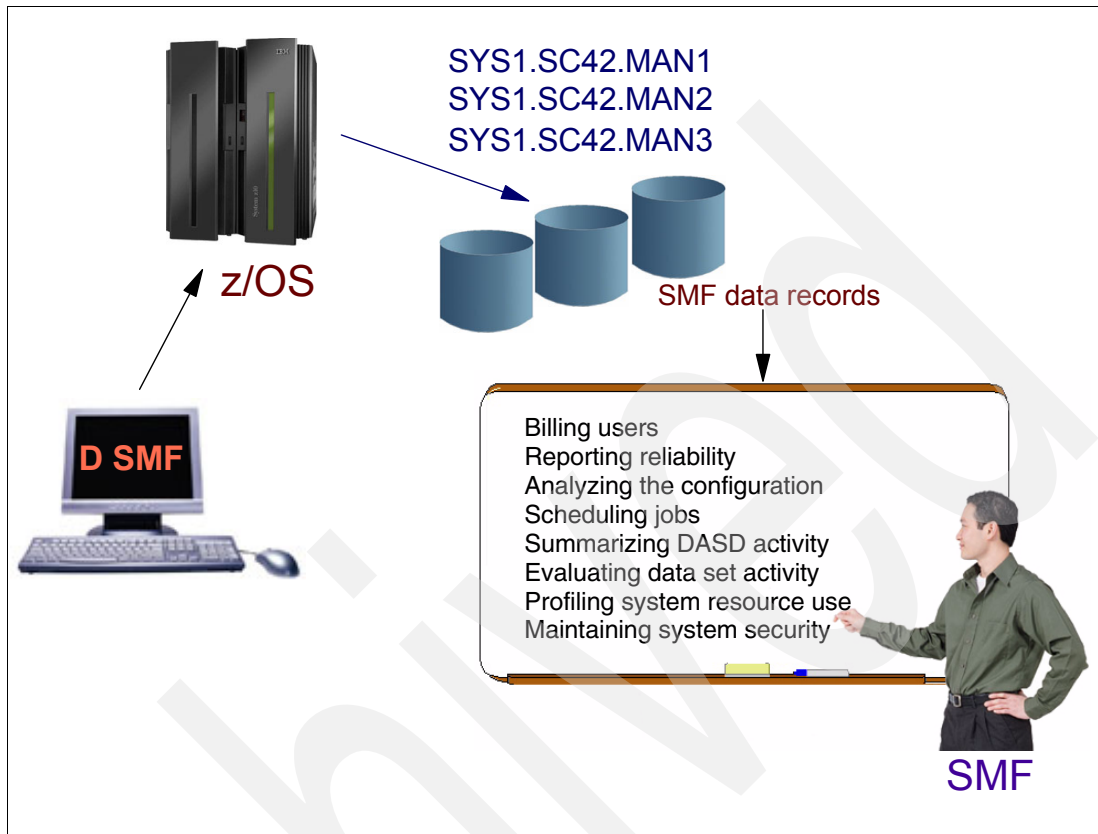


Figure 3-32 System Management Facility

System Management Facility

The z/OS system collects statistical data for each task when certain events occur in the life of the task. The System Management Facility (SMF) formats the information that it gathers into system-related (or job-related) records. System-related SMF records include information about the configuration, paging activity, and workload. Job-related records include information about the CPU time, SYSOUT activity, and data set activity of each job step, job, APPC/MVS transaction program, and TSO/E session. SMF data is written to the SYS1.MANx data sets.

By issuing the **D SMF** command, you can see a response similar to that shown in Figure 3-33. It identifies three SMF data sets, one of which is ACTIVE, and it is this active SMF data set (SYS1.SC42.MAN1) that is currently storing the system- and task-related performance data. When this data set fills up, the system will automatically start writing to the first available ALTERNATIVE SMF data set. The data that is stored in SYS1.SC42.MAN1 will be dumped to a data set, either on tape or DASD, using the IFASMFDD utility, which will also clear this data set and make it available for reuse.

```

IEE974I 14.38.54 SMF DATA SETS 613
      NAME                VOLSER SIZE(BLKS) %FULL STATUS
      P-SYS1.SC42.MAN1    MVS004   1200    17  ACTIVE
      S-SYS1.SC42.MAN2    MVS004   1200     0  ALTERNATE
  
```

Figure 3-33 Display of SMF data sets

3.31 Using SMF data records

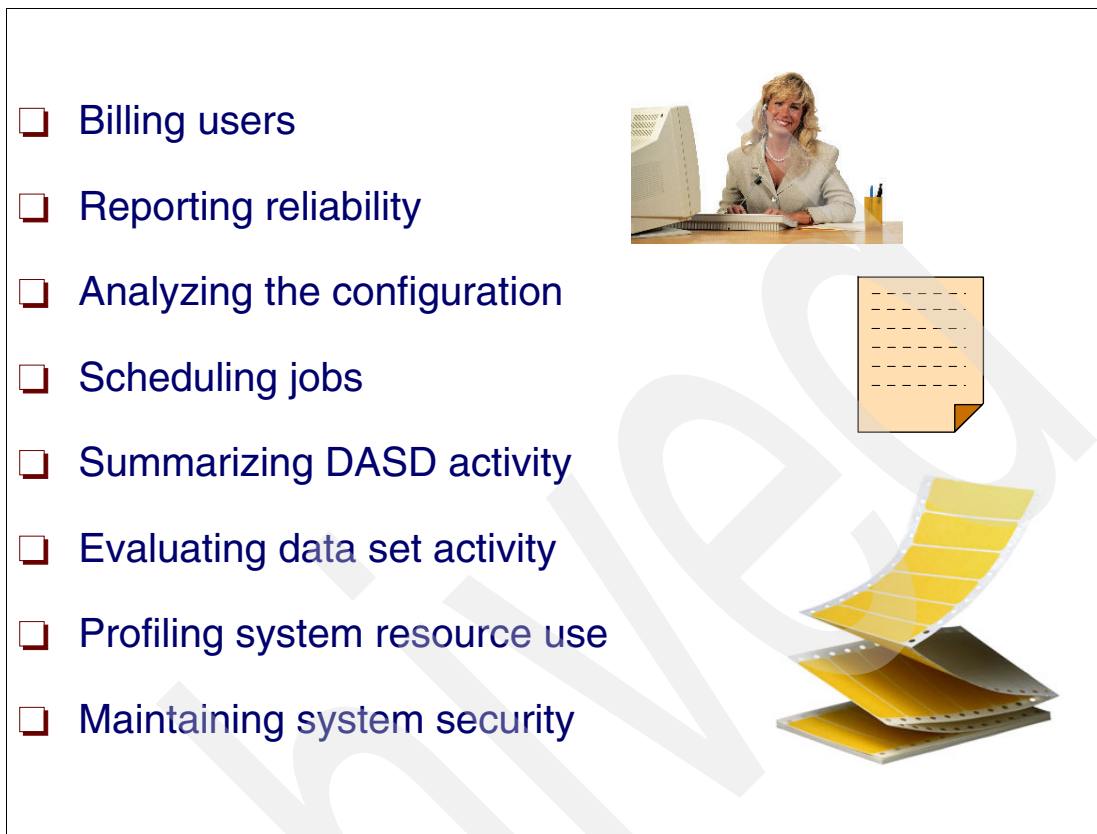


Figure 3-34 Using the SMF records

Using SMF records

The volume and variety of information in the SMF records enables installations to produce many types of analysis reports and summary reports. SMF formats the information that it gathers into system-related records or job-related records, as follows:

- ▶ System-related SMF records include information about the configuration, paging activity, and workload.
- ▶ Job-related records include information about the CPU time, SYSOUT activity, and data set activity of each job step, job, APPC/MVS transaction program, and TSO/E session.

Billing users

SMF reports data that installations can use as a basis for billing algorithms and reports.

Reporting reliability

SMF produces records at IPL time and at system shutdown. By examining these records and the last SMF record recorded before shutdown of the system, an installation can establish information over a given time period or reporting interval. This information includes how many IPLs, system uptime and downtime, and reasons for system failures.

Analyzing the configuration

SMF generates records that describe changes in the system configuration, such as at IPL for online devices, when devices are added or removed, or when a processor, channel path, storage device moves online or offline.

Scheduling jobs

Using SMF collected data, you can identify specific intervals when the problem program use of system resources is at an extremely high or low level. By studying the trends in this SMF data and the relationships among the trends operations management can establish and enforce its job scheduling procedures.

Summarizing DASD activity

SMF reports information about problem-program use of direct access volumes. This data can show operations personnel a possible need to examine problem-program use of direct access storage.

Evaluating data set activity

SMF produces several records that contain information about data set activity. By sorting these records by job name or user ID, an installation can produce a report of the data sets that were defined, modified, or deleted by problem programs during a specified interval. Such a report might be useful in a backup situation, especially when critical data sets have been unintentionally altered or destroyed.

Profiling system resource use

Most SMF records contain general identification fields such as the job name, step name, programmer name, reader start time, and reader start date. By sorting and summarizing SMF data according to these types of fields, an installation can create reports or profiles that show each batch job, job step, and TSO/E session's use of system resources such as CPU time, storage, paging facilities, I/O devices, service units, and programming languages.

Maintaining system security

An installation can use RACF SMF records to analyze the following items:

- ▶ Track the total use of a resource.
- ▶ Identify the resources that are repeated targets of detected unauthorized attempts to access them.
- ▶ Identify the users who make detected unauthorized requests.
- ▶ Track the activity of a particular user.
- ▶ Perform operator command auditing.

RACF provides options that allow your installation to control access to resources and, through SMF records, audit your security controls.

Note: Many of these options significantly increase the number of SMF records your system generates.

3.32 Importance of SMF records

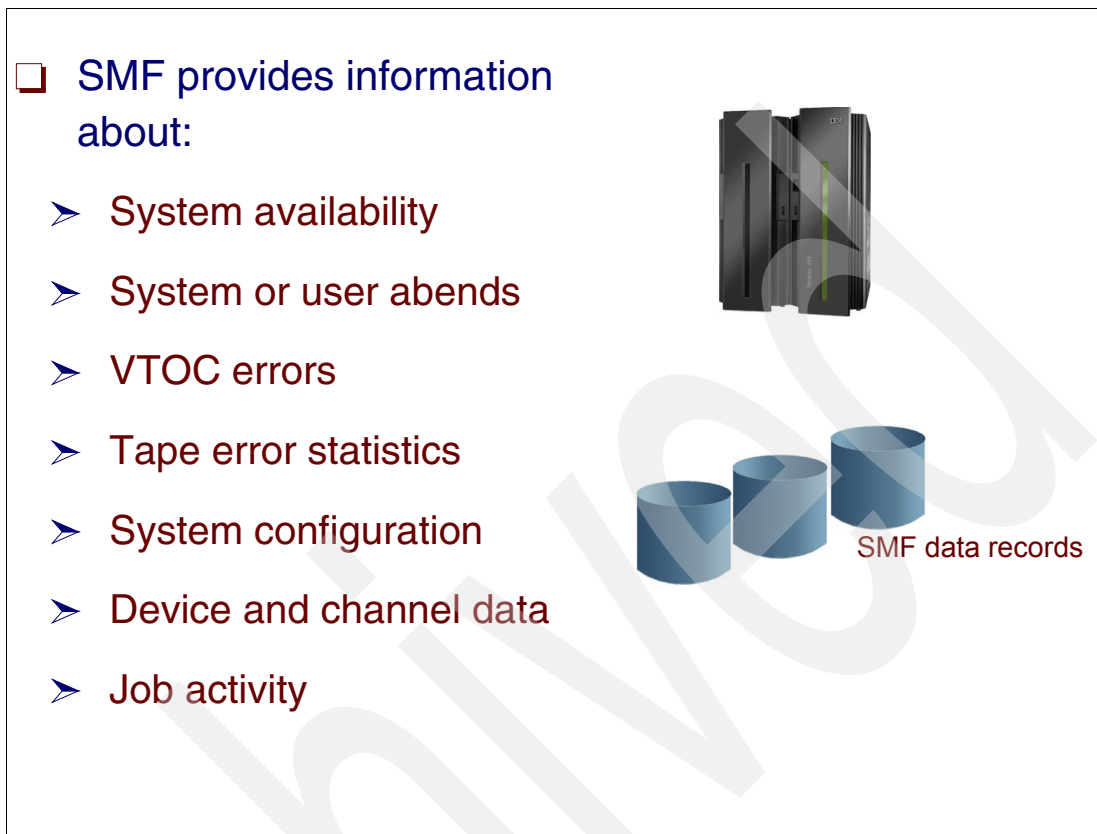


Figure 3-35 Importance of SMF records

The importance of SMF records

It is important to discuss information that can be obtained by reviewing the SMF records which create a database that can be used to understand trends and enable you to prepare for future workload and threshold problems. The volume and variety of information in the SMF records enables installations to produce many types of analysis reports and summary reports. For example, by keeping historical SMF data and studying its trends, an installation can evaluate changes in the configuration, workload, or job scheduling procedures. Similarly, SMF data can be used to determine system resources wasted because of problems, such as inefficient operational procedures or programming conventions.

System availability

SMF produces records at IPL time and also when an operator enters a **HALT EOD** command preceding the scheduled shutdown of the system. By examining these records and the last SMF record recorded before shutdown of the system, an installation can establish the following information for a given time period:

- ▶ Reporting interval number of IPLs
- ▶ System uptime and system downtime
- ▶ Number of scheduled and unscheduled stoppages and the approximate amount of scheduled and unscheduled down time
- ▶ Reasons for system failure

System or user abends

SMF reports a system or user abend (abnormal end task) code for each job (and job step) that abends. By tracking the codes issued by operational procedures (such as codes 122 and 222 for operator cancels), an installation can account for any loss of CPU time due to job reruns. More generally, a summary of the abend codes by program name or code allows an installation to determine which programs are abending frequently and which codes are occurring most often. This information can show the need for software error corrections, JCL revisions, or better operating instructions.

DASD VTOC errors

SMF record type 19 has a VTOC indicator bit that the system sets if there is a failure when updating a volume table of contents (VTOC). By checking the setting of this bit, operations personnel can identify any VTOCs that might have missing tracks or overlapping data sets.

Tape error statistics

SMF record type 21 provides tape error statistics such as the number of temporary read and write errors, permanent read and write errors, noise blocks, erase gaps, and cleaner actions. By sorting and summarizing these error statistics by tape volume (or tape unit), operations personnel can identify volumes that might need reconditioning or replacement, or point out tape drives that might require cleaning or maintenance.

System configuration

SMF generates records that describe changes in the system configuration, as follows:

- ▶ At IPL for online devices (types 0, 8, 19, and 22)
- ▶ When a device is added to the configuration (types 9 and 10)
- ▶ When a device is removed from the configuration (type 11)
- ▶ When a processor, channel path, or storage device moves online or offline (type 22)

Device and channel data

From SMF records, an installation can obtain the total problem program EXCP counts by device and by channel over a given reporting period. An installation can combine the data in the SMF step termination records to report the number of devices per device type that problem programs used during specified intervals. By using this report with the RMF device activity records (type 74), an installation can identify periods of the day when the percentage of problem program device use was exceptionally high or low.

Job activity

The SMF job initiation and termination record contains the start and stop times of each batch job, job step, TSO/E session, APPC/MVS transaction program, and started task. Using these times, an installation can determine the jobs that are running during the same interval. Evaluation of this data can assist with optimizing system performance by rescheduling part of the workload.

Volume mounting

SMF writes a type 19 record whenever a volume that is defined by a DD statement is demounted. Summarizing these records by volume gives an installation an indication of its direct access volume mounting activity for problem programs. For fragmented volumes, periodic analysis of the SMF type 19 records can be useful in identifying direct access volumes whose unallocated space is fragmented.

3.33 Importance of SMF records

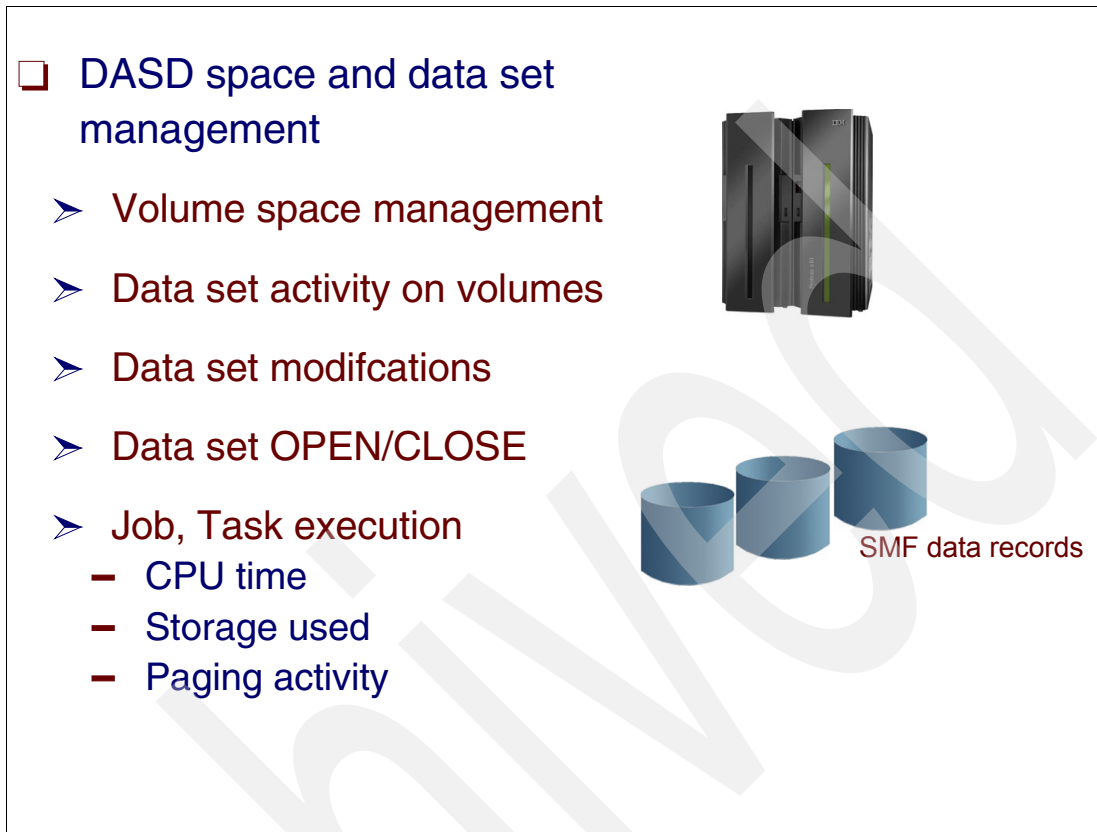


Figure 3-36 Importance of SMF records

DASD and data set management

Evaluating Data Set Activity SMF produces several records that contain information about data set activity. These records, which include types 4, 14, 15, 17, 18, 30, and 34, can help the installation to answer the following questions:

- ▶ What is the average data set size for both tape and direct access devices?
- ▶ Is the number of multivolume data sets significantly large?
- ▶ What percentage of all data sets are permanent? What percentage are temporary?
- ▶ What percentage of all temporary data sets does virtual input output (VIO) control?

Volume space management

An installation can identify the volumes that might need reorganization by examining the relationship of the following SMF fields:

- ▶ The number of unallocated cylinders and tracks.
- ▶ The number of cylinders and tracks in the largest unallocated extent.
- ▶ The number of unallocated extents.
- ▶ For multiple extents on volumes, by checking the number of extents field in the UCB section of SMF type 14 and 15 records, an installation can identify direct access data sets that have exceeded their primary allocation and have used secondary allocation(s). Although useful, secondary allocation can affect system performance and fragment the space on direct access volumes.

Data set activity on volumes

SMF produces several records that contain information about data set activity. These records, which include types 4, 14, 15, 17, 18, 30, and 34 can help an installation to answer questions about this activity:

- ▶ For data set modifications, SMF generates a record each time a user takes the following actions:
 - Scratches a non-VSAM data set (type 17)
 - Renames a non-VSAM data set (type 18)
 - Updates a VSAM data set (type 60)
 - Defines a catalog entry for the integrated catalog facility (type 61)
 - Alters or renames a catalog entry for the integrated catalog facility (type 66)
 - Defines or alters a VSAM catalog entry (type 63)
 - Deletes a catalog entry for the integrated catalog facility (type 65)
 - Deletes a VSAM catalog entry (type 67)
 - Renames a VSAM catalog entry (type 68)
- ▶ For Open/Close activity, SMF writes a type 14 or 15 record whenever a data set is closed or processed by EOV. The installation can determine how many times EOV closed or processed a given data set by counting the number of type 14 and 15 records.
- ▶ By examining the block size and logical record length fields recorded in the SMF type 14 and 15 records, an installation can identify data sets that the system is processing with ineffective blocking factors.

SMF records general identification

Most SMF records contain general identification fields such as the job name, step name, programmer name, reader start time, and reader start date. By sorting and summarizing SMF data according to these types of fields, an installation can create reports or profiles that show each batch job, job step, and TSO/E session's use of system resources such as:

- ▶ CPU time
- ▶ Storage
- ▶ Paging facilities
- ▶ I/O devices
- ▶ Service units

3.34 Recording SMF data records

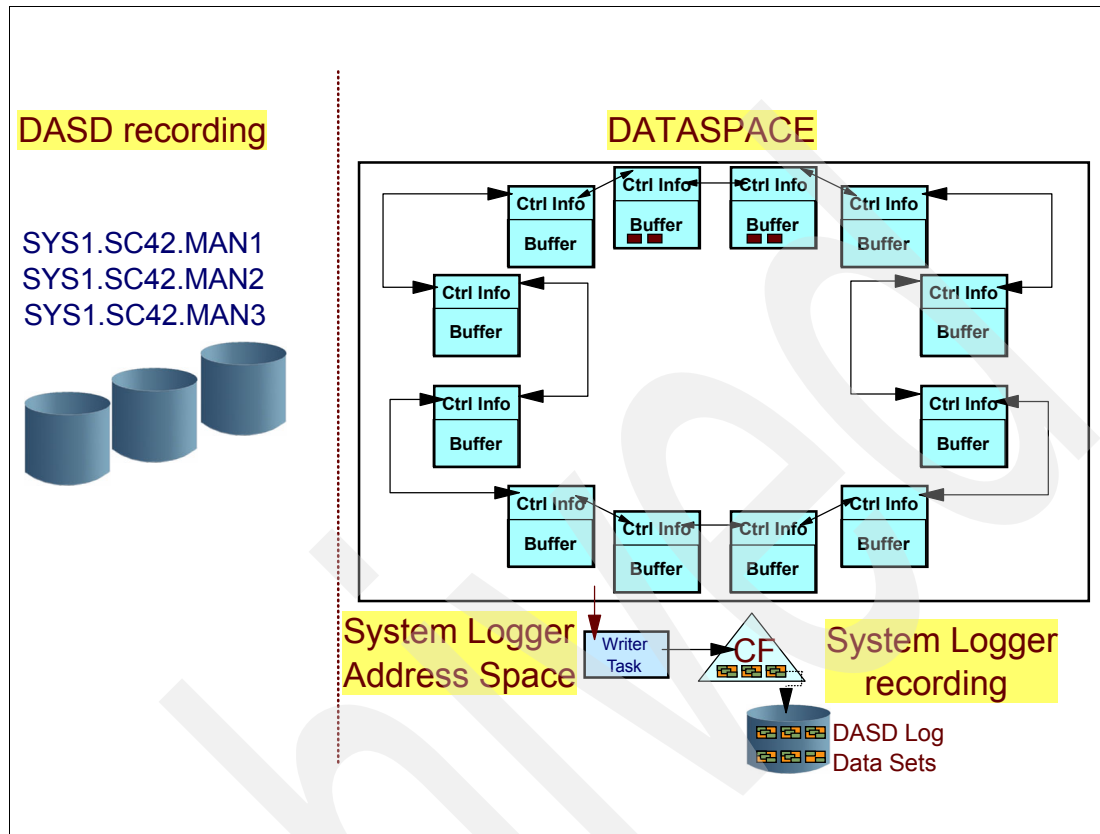


Figure 3-37 Method to record SMF records

Recording SMF records

Installations can decide whether to use SMF data sets (shown on the left section of Figure 3-37) or to write SMF records to log streams (shown on the right section of Figure 3-37), or both.

Note: SMF recording to log stream data sets became available with z/OS V1R9.

DASD SMF data sets

When using SMF data sets, SMF will write records to the SMF data sets you allocate. The size of the data that the system can write to SMF data sets is constrained by the VSAM control interval size. SMF can only write one control interval at a time.

System Logger SMF data sets

Beginning with z/OS V1R9, if you use SMF logging, SMF writes records to the system logger-managed log streams that you set up. SMF can write much larger chunks of data to the log stream than it can to SMF data sets. This has the potential of making writing SMF records faster, which can mean collecting more data.

SMF logging also allows you to group different SMF record types into different log streams defined for various purposes. For example, perhaps you want a log stream for job-related SMF data and another log stream for performance SMF data. Furthermore, you can write a single record to multiple log streams.

3.35 SMF recording to DASD (SYS1.MANx)

- ❑ SMF records in SMF data sets
 - Select DASDs that can handle the volume of data
 - Consider the amount of SMF data to be written
 - The size of SMF buffers
 - Specify buffers in SMFPRMxx parmlib member
 - BUFSIZMAX
 - BUFUSEWARN
 - NOBUFFS

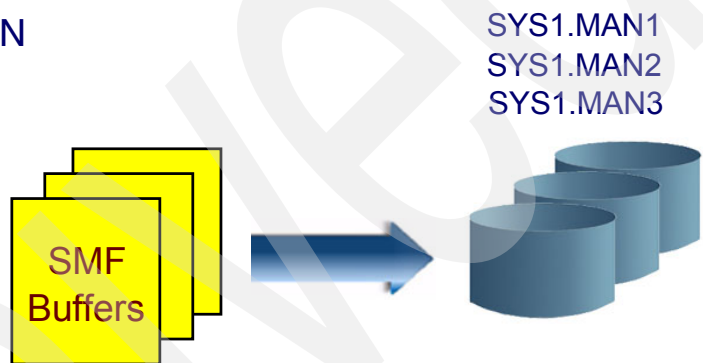


Figure 3-38 SMF recording to DASD (SYS1.MANx)

SMF recording to DASD

To record SMF records in SMF data sets, an installation must allocate direct access space and catalog the SMF data sets. It is good practice to catalog the SMF data sets in the master catalog. SMF is to have a minimum of two data sets for its use, and it is useful to run with a minimum of three SMF data sets to ensure availability.

Select DASDs that can handle the volume of data that SMF generates at your installation. Create at least two SMF data sets to store the system-related and job-related information that SMF collects. SMF data sets are to be RACF-protected. Each one is identified by a 1-44 character data set name. SMF data-gathering routines fill the data sets one at a time. While the gathering routines write records on one data set, SMF can write out or clear the others. SMF continues to write records as long as it can find an empty inactive data set when the active data set becomes full.

SMFPRMxx buffer parameters

If the I/O rate for a device is too slow, SMF places the data it generates in buffers. The buffers will eventually fill, which can result in lost data. Several factors, such as the specific system configuration, the amount of SMF data to be written, the size of SMF buffers (the control interval size), and your installation's report program requirements, determine which device type is most efficient for a particular installation.

BUFSIZMAX (nnnnM) (1G)

This specifies the maximum amount of storage that SMF can allocate for SMF record data buffering purposes. This parameter applies only when recording to SMF data sets. It does not apply when recording to SMF log streams.

- ▶ BUFSIZMAX value areas are defined with nnnnM for megabytes or 1G for one gigabyte. For example, to request 1 gigabyte, specify BUFSIZMAX(1G) or BUFSIZMAX(1024M). To request 128 megabytes, specify BUFSIZMAX(128M).
- ▶ The BUFSIZMAX value can be specified during an IPL in parmlib member SMFPRMxx, and the value can be set higher or lower by using the T **SMF** or **SETSMF** commands.

BUFUSEWARN (dd)

This specifies the overall buffer warning level percentage when SMF will start issuing message IEE986E. When the amount of in-use buffer percentage falls below the BUFUSEWARN value minus 5 (the default is 20%), message IEE986E will be deleted. This parameter applies only when recording to SMF data sets. It does not apply when recording to SMF log streams.

When SMF is using this percent of overall buffer space (see BUFSIZMAX), message IEE986E will be issued. As each additional or incremental (8M) SMF buffer is used to buffer SMF record data, SMF will issue an updated instance of IEE986E with the new buffer storage percent in use indication. As in-use SMF buffers are no longer needed, the buffers are removed from the in-use chain. After eligible buffers are removed, an updated instance of message IEE986E will be issued to reveal the changed (reduced) buffer storage percent in use indication. When the overall SMF buffer in use percentage drops to 5% below the BUFUSEWARN value, SMF will DOM message IEE986E.

NOBUFFS { (MSG) | { (HALT) }

This specifies the system action when the SMF address space has run out of buffer space.

- ▶ **MSG** specifies that the system is to issue a message and continue processing; SMF data is lost until buffer storage is again available.
- ▶ **HALT** specifies that the system is to enter a restartable wait state. HALT means that no SMF data is lost.

3.36 SMF recording to SYS1.MANx

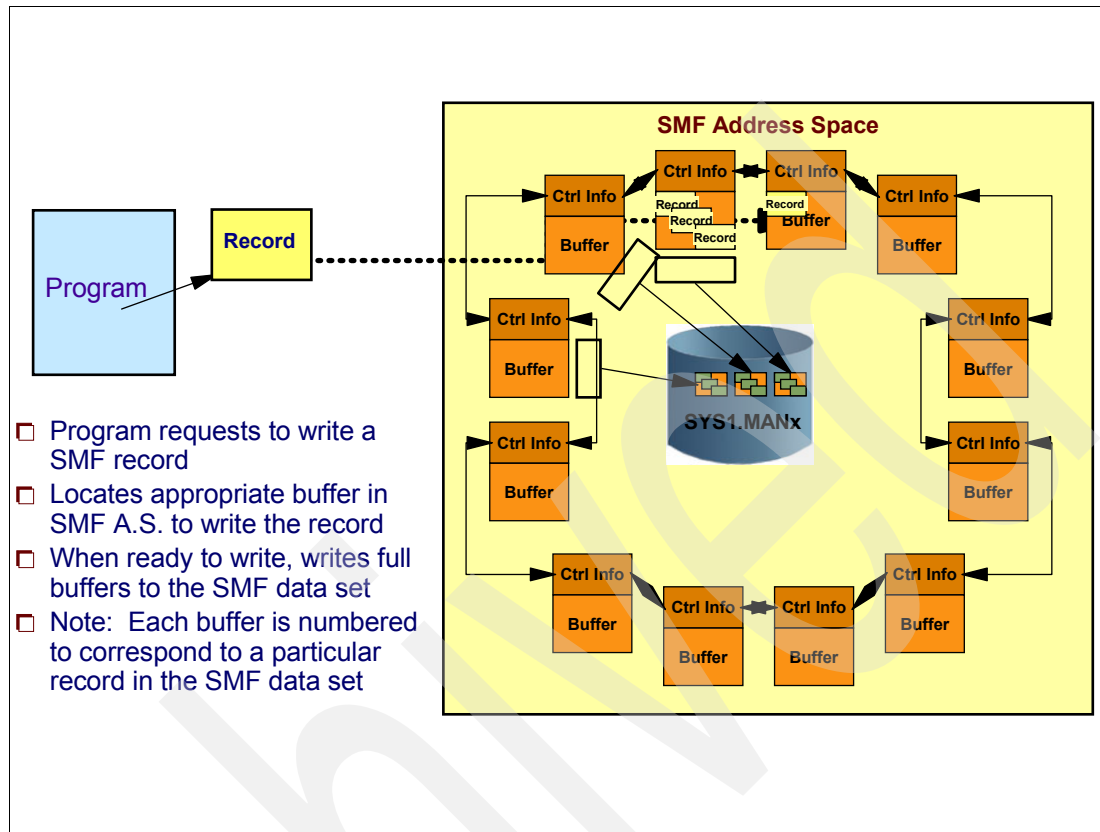


Figure 3-39 SMF recording to SYS1.MANx

SMF recording

When a subsystem or user program wants to write an SMF record, it invokes the SMF record macro SMFEWTM. This macro takes the user record and invokes SMF code to locate an appropriate buffer in the SMF address space and copy the data there, as shown in Figure 3-39. If the record is full, another SMF program is scheduled to locate full SMF buffers and write them to the SYS1.MANx data set. Each buffer is numbered to correspond to a particular record in the SMF data set. This allows the records to be written in any order and to place them correctly in the data set.

Although this is not shown in Figure 3-39, after all records have been written and the SYS1.MANx data set is full, SMF switches to a new SYS1.MANx data set and marks the full one as DUMP REQUIRED. That data set cannot be used again until it is dumped and cleared. Scheduling the SMF dump program must be done in a timely manner to ensure that the SMF MANx data set is returned to use as soon as possible to ensure that no data is lost due to an "all data sets full" condition.

Current implementation situations

The current implementation deficiencies are addressed by the support with SMF recording to the System Logger. Following are current situations with SMF recording to SYS1.MANx data sets:

- Several SYS1.MANx data sets can be defined. SMF records can be lost if all the SYS1.MANx data sets are filled up and not dumped, or during the SMF switch data set.

- ▶ There are situations when that system can generate a significant amount of SMF records, exceeding the system capacity to write them into the SYS1.MANx data sets.
- ▶ The current implementation of SMF can lose data if writes are being held up, when all SYS1.MANx data sets have become full or across SMF data set switch processing.
- ▶ In addition to the possibility of losing data when recording, the SMF dump program is required to read and write every record to move the data to archive data sets, where it can then be processed by other programs (such as for sorting), or by the SMF dump program again to further filter and partition the data for use. This can result in the data being read by the SMF dump program several times, as it is read and copied for use by the various exploiters of SMF data.
- ▶ The SMF records are written in any order, so they need to be sorted for post processing. In a sysplex environment, it is necessary to merge all system information to obtain a sysplex-wide analysis.
- ▶ Many installations have already set up automation to dump the SMF data sets using IFASMFDP as a postprocessor and they are satisfied with this function. This support continues the same. However, for installations that need more SMF data record write capacity, or for those whose SMF records are lost during a data set switch, it is unacceptable.

3.37 SMF on System Logger

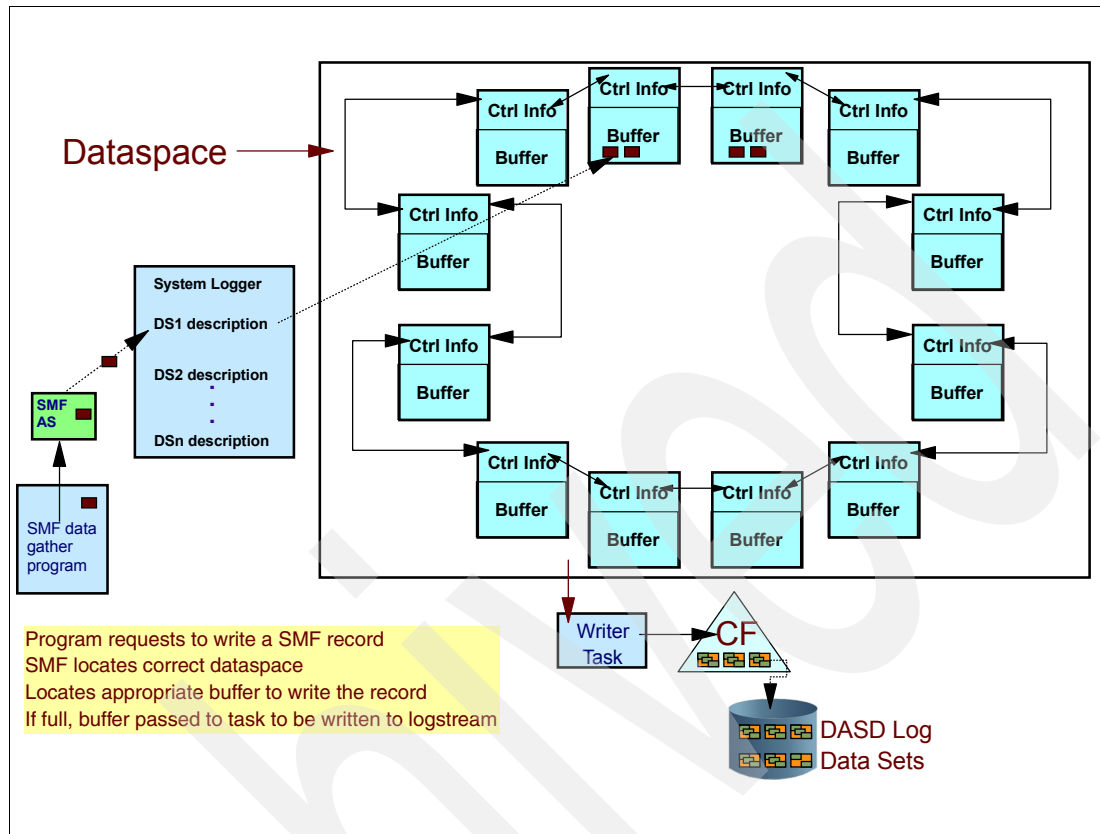


Figure 3-40 SMF on System Logger

SMF records on System Logger

z/OS V1R9 first introduced an additional capability to write SMF records to log streams managed by System Logger. With this new capability you can define several log streams for several groups of SMF records. You can define one log stream to write only the RMF records types, so during postprocessing they are already isolated. When you define one log stream, you must define the SMF records type that will be written to this log stream. You can define a default log stream to write all the remaining SMF records types not defined to a specific log stream.

Note: You can create as many dataspace as needed to create SMF log streams.

Recording to log streams

When recording to log streams, as shown in Figure 3-40, subsystems or user programs still invoke the SMFEWMT macro to take the user record and invoke SMF code. However, instead of locating a buffer in SMF private storage, SMF locates a data space corresponding to the user's record type and log stream where the record will be written. A buffer with space to hold the record is located and the record is copied there. When the record is full, a writer task is posted.

Unlike the scheduled approach in SMF data set recording, this task is already started and ready to write. In addition, writes to System Logger are done at memory-to-memory speeds,

with System Logger accumulating numerous records to write out, resulting in improved access not possible with current SMF data set recording.

Using a data space to hold the records for a given log stream allows a full 2 GB of pageable memory to be used for buffering records in the event of delays in log stream writing in System Logger. This allows more data to be buffered than with SMF data set recording, which is limited to the amount of available private storage in the SMF address space.

The benefit in all this is that you can write more data faster, with more functionality. The System Logger was created to handle large volumes of data. With minimal SMF switch processing and no record numbering schemes to maintain, this eliminates the switch SMF command bottleneck.

Note: The use of log streams for SMF Data is optional. Existing SYS1.MANx function continues to exist for installations satisfied with this functionality.

SMF with System Logger improvements

This new support provides the following functions and improvements:

- ▶ System Logger can improve the write rate and increase the volume of data that can be recorded.
- ▶ System Logger utilizes modern technology such as the Coupling Facility and Media Manager to write more data at much higher rates than the current SMF SYS1.MANx data set allows.
- ▶ Data management is improved by enhancing SMF to record its data to multiple System Logger log streams, based on record type. The record data is buffered in data spaces, instead of the SMF address space private storage, thus allowing increased buffering capacity.
- ▶ Keywords are provided on the OUTDD keyword of the dump program that allows data to be read one time and written many times. By recording different records to different log streams, SMF dump processing is also improved because a dump program per log stream can be submitted, with each being more efficient, because fewer records are read and ignored. This reduces processing time because there is less filtering needed by the dump program
- ▶ One SMF record type can be written to more than one log stream. By selecting log streams based on record type, the data can be partitioned at the point of its creation, resulting in less reprocessing by the SMF dump program, which means less data read per dump program instance.
- ▶ For each SMF log stream, a database is created as a buffer in the SMF ASID, so each buffer is 2 GB. Within SMF, each databases will have a task dedicated to writing its data to a particular log stream, increasing the rate at which you can record data to the System Logger.

3.38 Where System Logger stores data

- ❑ Application sends log data to the System Logger
 - Log stream data can initially be stored on DASD, in what is known as a DASD-only log stream, or...
 - Log stream data can be stored in a Coupling Facility (CF) in what is known as a CF-structure log stream
- ❑ CF log stream - interim storage for log data is in CF list structures
 - Allows for exploiters on more than one system to write log data to the same log stream concurrently
- ❑ DASD-only log stream - interim storage for log data is contained in a dataspace in the z/OS system
 - DASD-only log streams can only be used by exploiters on one system at a time

Figure 3-41 System Logger log streams

Where System Logger stores its data

When an application passes log data to the System Logger, the data can initially be stored on DASD, in what is known as a DASD-only log stream. Alternatively, it can be stored in a Coupling Facility (CF) in what is known as a CF-Structure log stream. The major differences between these two types of log stream configurations are the storage medium System Logger uses to hold interim log data, and how many systems can use the log stream concurrently, as explained here.

- ▶ In a CF log stream, interim storage for log data is in CF list structures. This type of log stream supports the ability for exploiters on more than one system to write log data to the same log stream concurrently.
- ▶ In a DASD-only log stream, interim storage for log data is contained in a data space in the z/OS system. The data spaces are associated with the System Logger address space, IXGLOGR. DASD-only log streams can only be used by exploiters on one system at a time.

Note: Your installation can use just Coupling Facility log streams, just DASD-only log streams, or a combination of both types of log streams. The requirements and preparation steps for the two types of log streams are somewhat different. See “Setting Up the System Logger Configuration” in *z/OS MVS Programming: Authorized Assembler Services Guide*, SA22-7608 and *Systems Programmer's Guide to: z/OS System Logger*, SG24-6898.

3.39 Customizing SMF

- ❑ Use SMFPRMxx parmlib member
 - Select which SMF records are to be recorded
 - Records have subtypes
 - Specify data set names and other options
 - Other options
- ❑ SMF record types
- ❑ SMF installation exits
- ❑ Using DASD for SMF data sets
- ❑ Using SMF dump exits
 - IEFU29 and IEFU29L

Figure 3-42 Customizing SMF

Customizing SMF

Setting up SMF requires your installation to decide what kind of records it wants SMF to produce or what information it wants SMF to gather. Then you can make decisions about how to set SMF up to meet these requirements.

An installation has several ways to customize SMF to meet its needs:

- ▶ SMF parameters on the SMFPRMxx parmlib member
- ▶ Installation-written exit routines
- ▶ Operator commands

SMPRMxx parmlib member

The SMF parmlib parameters allow you to perform the following tasks:

- ▶ Record status changes
- ▶ Pass data to a subsystem
- ▶ Select specific records
- ▶ Specify data set names
- ▶ Specify the system identifier to be used in all SMF records
- ▶ Select specific subtypes
- ▶ Perform interval accounting
- ▶ Collect SMF statics
- ▶ Perform TSO/E command accounting
- ▶ Perform started task accounting

SMF record types

SMF records are selected by specifying either the type desired (or the types not desired) with the TYPE or NOTYPE option of the SYS or SUBSYS parmlib parameter. If any one of record types 14, 15, 17, 62, 63, 64, 67, or 68 is specified with the TYPE option, data is collected for *all* records. Likewise, if either record type 19 or 69 is specified with the TYPE option, data is collected for *both* records. However, only those records that are selected by TYPE or NOTYPE request are written to the SMF data set.

The TYPE option of the SYS and SUBSYS parameter provides inner parentheses to indicate the subtypes to be collected. If subtype selection is not used, the default is all subtypes. The following example illustrates how the TYPE option is to be used to record subtypes 1, 2, 3, 5, 6, 7, and 8 for the type 30:

```
SYS(TYPE(30(1:3,5:8))) or SUBSYS(STC,TYPE(30(1:3,5:8)))
```

SMF installation exits

Installation-written exit routines IEFU83, IEFU84, and IEFU85 (SMF writer) and IEFACTRT (termination) can control which records are to be written to the SMF data set. After inspecting an SMF record, these routines return a code to the system indicating whether the record is to be written to the SMF data sets.

SMF data sets

To record SMF records in SMF data sets, an installation can allocate direct access space and catalog the SMF data sets. A good practice is to catalog the SMF data sets in the master catalog. SMF is to have a minimum of two data sets for its use, and it is more desirable to run with a minimum of three SMF data sets to ensure availability.

When selecting DASD to handle the volume of data that SMF generates, if the I/O rate for a device is too slow, SMF places the data it generates in buffers. The buffers will eventually fill, which can result in lost data. Several factors, such as the specific system configuration, the amount of SMF data to be written, the size of SMF buffers (the control interval size), and your installation's report program requirements, determine which device type is most efficient for a particular installation.

SMF dump exits (IEFU29)

The SMF dump exit IEFU29 is invoked when the current recording data set cannot hold any more records, because the SMF writer routine automatically switches recording from the active SMF data set to an empty SMF data set. This exit is also invoked when the writer switches recording data sets as a result of the **SWITCH SMF** command. A return code from this exit routine indicates whether a message that the SMF data set requires dumping is to be suppressed or not.

To allow the system to invoke IEFU29, define the exit in the SMF parmlib member (SMFPRMxx). Specify IEFU29 on the EXITS option of the SUBSYS parameter for the STC subsystem. If your installation chooses not to define a SUBSYS parameter for STC, you can specify IEFU29 on the EXITS option of the SYS parameter.

SMF dump exits (IEFU29L)

The SMF dump exit IEFU29L allows you to initiate the archiving of SMF data from a log stream. IEFU29L is invoked using the **SWITCH SMF** command. You can use the system logger to manage how long you retain the SMF log stream data and to automatically offload the log stream data to VSAM linear DASD data sets, so you might not need to use IEFU29L to drive archiving the SMF log stream data.

3.40 Which SMF records to record

- ❑ SMFPRMxx parmlib member
 - Choose DASD data sets or System Logger log streams
 - RECORDING(DATASET | LOGSTREAM)
 - Selecting records to write for DASD data sets:
 - Specifying either the type or no type with the TYPE or NOTYPE option of the SYS or SUBSYS parameter
 - Selecting records to write for System Logger data sets:
 - DEFAULTLSNAME(IFASMF.DEFAULT)
 - LSNAME(IFASMF.PERF,TYPE(30,89))
 - LSNAME(IFASMF.JOB,TYPE(30))
 - RECORDING(LOGSTREAM)

Figure 3-43 SMF records to record

Select DASD data sets or System Logger log streams

You must specify whether you want to write SMF records to MANx data sets or log streams. You can specify both SMF data sets and log streams in one SMFPRMxx member and then set either RECORDING(DATASET) or RECORDING(LOGSTREAM) on the SETSMF command to switch between data set and log stream recording. The default is: RECORDING(DATASET).

SMF records to record for DASD data sets

You can select the SMF records you want to write by specifying either the type desired (or the types not desired) with the TYPE or NOTYPE option of the SYS or SUBSYS parmlib parameter. If any one of record types 14, 15, 17, 62, 63, 64, 67, or 68 is specified with the TYPE option, data is collected for all of those record types. However, only records selected by a TYPE or NOTYPE request are written to the SMF data set, as shown here.

```
TYPE      {aa,bb(cc)      }
NOTYPE    ({aa,bb:zz     })
          {aa,dd(cc:yy),...}
          {aa,bb(cc,...)  }
```

TYPE This specifies the SMF record types and subtypes that SMF is to collect. aa, bb, dd, and zz are the decimal notations for each SMF record type. cc and yy are the decimal notations for the SMF record subtypes. A colon (:) indicates the range of SMF record types (bb through zz) to be recorded or the range of subtypes (cc

through yy for SMF record type dd) to be recorded. You can select SMF record subtypes on all SMF record types, and on user records.

NOTYPE This specifies that SMF is to collect all SMF record types and subtypes except those specified. aa, bb, and zz are the decimal notations for each SMF record type. cc and yy are the decimal notations for each subtype. A colon indicates the range of SMF record types (bb through zz) or the range of subtypes (cc through yy for SMF record dd) that are not to be recorded.

When RECORDING(DATASET) is specified, the default is SYS1.MANX and SYS1.MANY.

SMF records to record for System Logger data sets

The SMFPRMxx parmlib member parameters to use System Logger recording are listed here.

DEFAULTLSNAME(logstreamname)

LSNAME (logstreamname,TYPE({aa,bb}|{aa,bb:zz} | {aa,bb:zz,...}))

RECORDING(LOGSTREAM)

DEFAULTLSNAME(logstreamname)

This optional parameter specifies the default log stream name you want to use when you write SMF records to a log stream, except for the record types specified on LSNAME parameters. When you specify DEFAULTLSNAME (and do not override it with the LSNAME parameter), records will be queued and written to the default log stream name specified; for example:

DEFAULTLSNAME(IFASMF.DEFAULT)

LSNAME(logstreamname,TYPE({aa,bb}|{aa,bb:zz} | {aa,bb:zz,...}))

This optional parameter allows you to specify the log stream in which you want to record particular SMF record types on the TYPE subparameter.

The log stream name (logstreamname) must be composed as shown here.

- ▶ The first seven characters must be IFASMF. (which is the qualifier).
- ▶ You must have a minimum of 8 characters.
- ▶ You must have a maximum of 26 characters.

Note: All log stream names (logstreamname) must have the first name qualifier IFASMF. of seven characters in length.

The TYPE operand specifies the SMF record types that SMF is to collect to the specified log stream on the LSNAME parameter. aa, bb, and zz are the decimal notations for each SMF record type. You cannot specify subtypes on the TYPE subparameter for LSNAME. A colon indicates the range of SMF record types (bb through zz) to be recorded.

Value Range: 0-255 (SMF record types)

Default: TYPE (0:255) (all types)

When RECORDING(LOGSTREAM) is specified, there is no default.

3.41 System Logger log streams

- ❑ SMFPRMxx parmlib member sample definitions
 - DEFAULTLSNAME(IFASMF.DEFAULT)
 - LSNAME(IFASMF.PERF,TYPE(30,89))
 - LSNAME(IFASMF.JOB,TYPE(30,04))
 - RECORDING(LOGSTREAM)
- ❑ Possible to switch SMF recording type by command
 - SET SMF=xx
- ❑ Possible to set SMFPRMxx parameters by command
 - SETSMF parameter(value[,value]...)
- ❑ SWITCH SMF command

Figure 3-44 System Logger log streams and commands

System Logger log streams

If you specify the same record type on two or more different LSNAME parameters, the system writes the record to all specified log streams. For example, you can have an SMFPRMxx parmlib member with the following contents:

```
DEFAULTLSNAME(IFASMF.DEFAULT)
LSNAME(IFASMF.PERF,TYPE(30,89))
LSNAME(IFASMF.JOB,TYPE(30))
RECORDING(LOGSTREAM)
```

These definitions allow you to collect job-related SMF data in the JOB log stream, and performance-related SMF data in the PERF log stream. Record type 30 fits into both categories, so you can specify that it is written to both log streams.

Note: This arrangement can result in duplicate records being recorded because each LSNAME statement with the same record type (30) is written into another log stream.

```
DEFAULTLSNAME(IFASMF.DEFAULT)
LSNAME(IFASMF.PERF,TYPE(30,89))
RECORDING(LOGSTREAM)
```

These definitions result in record types 30 and 89 going to log stream IFASMF.PERF. All other record types will go to default log stream IFASMF.DEFAULT.

Using the SET SMF command

This command can be used in several ways, as explained here.

- ▶ To switch from using SYS1.MANx recording to System Logger recording

Change the RECORDING parameter in a new SMFPRMxx parameter, and then use the **SET SMF=xx** command to switch the system to using that SMFPRMxx parmlib member.

Note: To stop recording to a particular SMF log stream, create a new SMFPRMxx parmlib member and remove the log stream you no longer want to use. Then issue the **SET SMF=xx** command to switch to using that SMFPRMxx parmlib member.

- ▶ To switch your SMF recording method

Issue the **SETSMF RECORDING(DATASET | LOGSTREAM)** command to switch to the method you prefer. Note that this method requires PROMPT(LIST) or PROMPT(ALL) be specified in the active SMFPRMxx parmlib member to allow use of the SETSMF command, for example, if your current mode is SYS1.MANxx recording or option DATASET. Then issue the following command to start doing System Logger recording:

```
SETSMF RECORDING(LOGSTREAM)
```

SWITCH SMF command

If you are using SMF data sets to record SMF records, the **SWITCH SMF** command manually switches the recording of SMF data from one data set to another. The **SWITCH SMF** command also passes control to the IEFU29 dump exit, if one exists.

If you are using log streams to record SMF records, the **SWITCH SMF** command writes data from an SMF buffer to a log stream in preparation for running the dump program to dump SMF data. The **SWITCH SMF** command also passes control to the IEFU29L dump exit, if one exists.

3.42 Dumping SMF data sets

- ❑ Using SMF dump programs
 - Dumping SMF log streams - IFASMFDL
 - Dumping the SMF data sets - IFASMFDL
- ❑ Dumping SMF data sets (IFASMFDL)
 - Transfer contents of SMF data sets to another data set
 - Set up a JCL stream to clear and dump
 - Use CLEAR option to free space on MANx data sets
- ❑ Dumping SMF log streams (IFASMFDL)
 - Set up a JCL stream to dump
 - New RELATIVEDATE parameter

Figure 3-45 Dumping SMF data sets

Dumping SMF data sets (IFASMFDL)

When notified by the system that a full data set needs to be dumped, use the SMF dump program (IFASMFDL) to transfer the contents of the full SMF data set to another data set, and to clear the dumped data set so that SMF can use it again for recording data. The dump program copies the input data sets to the output data sets. During the copy process, the SMF data set dump program creates two SMF records and writes them to every output data set.

Note: SMF data sets cannot be shared across systems. Avoid having the SMF data set dump program be run from one system in an attempt to clear a SMF data set used by another system. The SMF dump program allows the installation to route various records to separate files and produce a summary activity report. Figure 3-46 on page 245 shows a sample job stream.

```
//STEP1 EXEC PGM=IFASMFDL,REGION=4M
//DUMPIN DD DSN=SYS1.SC42.MAN1,DISP=SHR
//DUMPALL DD DISP=(,CATLG),DSN=SMFDATA.BACKUP(+1),UNIT=SYSALLDA,VOL=SER=SMF002,
// SPACE=(CYL,(30,10),RLSE),DCB=SMFDATA.ALLRECS.GDGMODEL
//SYSIN DD *
INDD(DUMPIN,OPTIONS(ALL))
OUTDD(DUMPALL,TYPE(000:255))
```

Figure 3-46 IFASMFDL job stream to dump and clear the SMF data set

In the IFASMFD job stream to dump and clear the SMF data set shown in Figure 3-46 on page 245, the **INDD(DUMPIN,OPTIONS(ALL))** parameter indicates that the SMF data set will be dumped and cleared. To process only a DUMP request, set **OPTIONS(DUMP)**. To process only a CLEAR request, set **OPTIONS(CLEAR)**. The **OUTDD(DUMPALL,TYPE(000:255))** parameter indicates that SMF records in the range of 000 to 255 will be dumped (that is, everything will be dumped). If you only want to dump TYPE30 Subtype 1, 3, 4 and 5, code **TYPE(30(1,3:5))**. Alternatively, if you do not want to dump these records, code **NOTYPE(30(1,3:5))**.

Note: The way you dump SMF data depends on whether you use log stream recording or SMF data set recording. Both the dump program for SMF log streams (IFASMF DL) and for SMF data sets (IFASMF DP) support the same exits.

Dumping to log streams (IFASMF DL)

To dump an SMF log stream to archive the data to a permanent medium such as tape, or so that existing user-written analysis routines can run against the dump data sets, you can dump SMF log stream data by issuing the SWITCH SMF command. This command first dumps the SMF data in the buffers out to the log streams, and then passes control to the IEFU29L SMF log stream dump exit.

Operators can use the SMF log stream dump program IFASMF DL to dump the specified log stream data to dump data sets. The SMF log stream dump program, shown in Figure 3-47, dumps the contents of one or more log streams to sequential data sets on either tape or direct access devices. The SMF log stream dump program allows the installation to route various records to separate files and produce a summary activity report.

```
//DUMPX      JOB      MSGLEVEL=1
//STEP1     EXEC     PGM=IFASMF DL
//OUTDD1    DD      DSN=BASCAR.LSDEFLT.RECOR66,DISP=(NEW,CATLG,DELETE),
//          UNIT=3390,
//          SPACE=(CYL,(1,1),RLSE),DCB=(LRECL=32760,RECFM=VBS,BLKSIZE=0)
//SYSPRINT  DD      SYSOUT=A
//SYSIN     DD      *
LSNAME(IFASMF.MULTSYS.STREAM1,OPTIONS(DUMP))
OUTDD(OUTDD1,TYPE(0:255))
RELATIVEDATE (BYDAY,3,3)
WEEKSTART (MON)
SID(SY1)
/*
```

Figure 3-47 SMF log stream dump program JCL

Figure 3-47 requests records to be selected from the log stream that were recorded starting from three days before the current date, and ending at the third day, counting from that starting day. To illustrate further, if the JCL in figure executes on Monday, July 28, 2008 (Julian date 2008.210), the data in the log stream from Friday July 25th 2008 (Julian date 2008.207) to Sunday July 27th (Julian date 2008.209) will be dumped.

Note: The RELATIVEDATE parameter is new with z/OS V1R11 and specifies a date range based on the current day to be selected by the IFASMF DL program. The start date of the date range is calculated by going backward n time units measured by day, week, or month from the current day, week, or month. The end date of the date range is calculated by moving x time units forward from the start date. You can use the RELATIVEDATE option instead of the DATE option, and it is not compatible with the DATE parameter.

3.43 Dumping SMF records - log streams

- ❑ Archiving SMF log records
 - Dump SMF log stream data by issuing the SWITCH SMF command
- ❑ SMF log stream dump program
 - Route different records to separate files and produce a summary activity report
 - Dump SMF data in the buffers out to the log streams
 - Then pass control to the IEFU29L SMF log stream dump exit
- ❑ IFASMF DL example

Figure 3-48 Dumping log streams records

Dumping log stream records to archive

To dump an SMF log stream to archive the data to a permanent medium such as tape, or so that existing user written analysis routines can run against the dump data sets, you can dump SMF log stream data by issuing the **SWITCH SMF** command. This command first dumps the SMF data in the buffers out to the log streams, and then passes control to the IEFU29L SMF log stream dump exit. The operator can use the SMF log stream dump program IFASMF DL to dump the specified log stream data to dump data sets.

SMF log stream dump program

The SMF log stream dump program dumps the contents of one or more log streams to sequential data sets on either tape or direct access devices. The SMF log stream dump program allows the installation to route various records to separate files and produce a summary activity report.

Note: When you use SMF logging, you can write individual record types to multiple log streams if you prefer, producing duplicate records. If you are dumping several log streams that contain the same record types, the dump can contain duplicate records. In addition, if you dump a log stream that contains data from multiple systems, then coordinate the time zone of the recording systems and the dumping system.

IFASMF DL example

In the example shown in Figure 3-49, the three OUTDD statements refer to the three data sets defined in the JCL, and they specify the SMF record types to be written to each data set. The JCL is explained here.

- ▶ The DCB= keyword has been coded for the output data set defined by OUTDD2. Any block size 4096 or greater can be specified. Choosing a block size suitable for the device type being used will improve storage resource use. For this job, the data set specified by OUTDD1 will have a system-determined block size. The data set specified by OUTDD2 will have a block size of 32000.
- ▶ The LRECL= keyword has been coded for an output data set defined as OUTDD3. For this job, the data set specified by OUTDD3 will have an LRECL of 32760. For OUTDD1 and OUTDD2, the LRECL will default to 32767.
- ▶ The LSNAME parameters contain the names of three log streams to be dumped.
- ▶ The OUTDD parameters contain filters selecting the SMF record types to be dumped:
 - OUTDD1 specifies that you want to dump record types 0,2,10,15-30, and subtype 1 of record type 33 starting with those issued at 7:30 am and ending at 6:50 pm.
 - OUTDD2 specifies that you want to dump record types 10 through 255 from dates October 1, 2006 through November 30, 2006.
 - OUTDD3 specifies that you want to dump record types 10 through 255.
- ▶ The DATE parameter specifies, for those OUTDD statements which do *not* include the DATE subparameter, that data from January 1, 2006 through December 31, 2006 is to be written.
- ▶ The SID parameters specify that data will be dumped for systems 308A and 308B.

```
//IFASMF DL JOB accounting information
//STEP EXEC PGM=IFASMF DL
//OUTDD1 DD DSN=SMFREC.FEWYPES,DISP=(NEW,CATLG,DELETE)
//OUTDD2 DD DSN=SMF.TYPE10.TYPE255,DISP=(NEW,CATLG,DELETE),
//   DCB=BLKSIZE=32000
//OUTDD3 DD DSN=SMF.TYPE10.TYPE255B,DISP=(NEW,CATLG,DELETE),
//   DCB=LRECL=32760
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
   LSNAME (IFASMF.DEFAULT)
   LSNAME (IFASMF.PERF)
   LSNAME (IFASMF.JOB)
   OUTDD (OUTDD1,TYPE(0,2,10,15:30,33(1)),START(0730),END(1850))
   OUTDD (OUTDD2,TYPE(10:255)),DATE(2006274,2006334)
   OUTDD (OUTDD3,TYPE(10:255))
   DATE (2006001,2006365)
   SID (308A)
   SID (308B)
   END(2400) - DEFAULT
   START(0000) - DEFAULT
```

Figure 3-49 Sample job for dumping SMF log streams

Note: There can be any number of input (LSNAME) or output (OUTDD) parameters in the SMF log stream dump program. The log streams are dumped in reverse order. For example, in Figure 3-49, three log streams are specified. After the SMF log stream dump program is processed, the output files contain the records from log stream IFASMF.JOB first, IFASMF.PERF next, followed by the records from IFASMF.DEFAULT.

3.44 Dumping selective SMF records

- ❑ Selective dumping of SMF records
 - Use IEFU29 dump exit
 - Use SMF dump data set program (IFASMFDP)
- ❑ Methods for dumping
 - Enter jobs specifying dump program
 - Hold on job queue
 - Submit JCL for dump program
- ❑ Recoding considerations
 - Choose options on SYS and SUBSYS parameters
 - TYPE or NOTYPE option

Figure 3-50 Selective dump of SMF records

Selective dump of SMF records

When the current recording data set cannot accommodate any more records, the SMF writer routine automatically switches recording from the active SMF data set to an empty SMF data set, and then passes control to the IEFU29 SMF dump exit. The operator is then informed that the data set needs to be dumped. SMF data sets cannot be shared across systems. Avoid having the SMF data set dump program be run from one system in an attempt to clear a SMF data set used by another system.

When notified by the system that a full data set needs to be dumped, operators use the SMF data set dump program (IFASMFDP) to transfer the contents of the full SMF data set to another data set, and to reset the status of the dumped data set to empty so that SMF can use it again for recording data.

Important: Use the IEFU29 SMF dump exit to run the SMF data set dump program. When the current recording data set cannot hold any more records, the SMF writer routine automatically switches recording from the active SMF data set to an empty SMF data set, and passes control to the IEFU29 dump exit. This avoids the need for operator intervention.

Methods for dumping

One method of running the SMF data set dump program is to enter jobs that specify the SMF data set dump program into the system, and hold them on the job queue until a dump is

required. Another method is to start a reader to an input stream containing the JCL for the SMF data set dump program. Figure 3-51 shows two sample procedures (DUMPX and DUMPY) for dumping the SMF data sets to a standard-labeled tape (VOL=SER=SMFTAP) with the operator **START** command.

```
//UPDATE JOB MSGLEVEL=1
//UPDATE EXEC PGM=IEBUPDTE,PARM=NEW
//SYSUT1 DD DSN=SYS1.PROCLIB,DISP=SHR
//SYSUT2 DD DSN=SYS1.PROCLIB,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSIN DD DATA
./ ADD NAME=DUMPX,LIST=ALL
//DUMPX PROC TAPE=192
//SMFDMP EXEC PGM=IFASMFDP
//DUMPIN DD DSNNAME=SMFDATA,UNIT=&TAPE,DISP=(MOD,KEEP),
// LABEL=(,SL),VOL=SER=SMFTAP
//SYSPRINT DD SYSOUT=A
./ ADD NAME=DUMPY,LIST=ALL
//DUMPY PROC TAPE=192
//SMFDMP EXEC PGM=IFASMFDP
//DUMPIN DD DSNNAME=SYS1.MANY,DISP=SHR
//DUMPOUT DD DSNNAME=SMFDATA,UNIT=&TAPE,DISP=(MOD,KEEP),
// LABEL=(,SL),VOL=SER=SMFTAP
//SYSPRINT DD SYSOUT=A
./ENDUP
/*
```

Figure 3-51 Sample procedures for dumping the SMF data sets

Figure 3-51 illustrates the following:

- ▶ The DCB= keyword has been coded for the output data set defined by OUTDD2. Any block size 4096 or greater can be specified. Choosing a block size suitable for the device type being used will improve storage resource use. For this job, the data set specified by OUTDD1 will have a system determined block size. The data set specified by OUTDD2 will have a block size of 32000.
- ▶ The LRECL= keyword has been coded for an output data set defined as OUTDD3. For this job, the data set specified by OUTDD3 will have an LRECL of 32760. For OUTDD2, the LRECL will default to 32767.

Recording considerations

Subtype selectivity for SMF records is an option of the TYPE or NOTYPE option on the SYS and SUBSYS parameters used for SMF recording. Subtype selectivity allows more flexibility in postprocessing records and helps control the amount of data stored in SMF data sets. The subtype selectivity function supports only records that use the standard SMF record header. The header requires the subtype field to be at offset 22 (decimal) and the “subtypes used” bit (bit position 1) of the system indicator byte at offset 4 (decimal) to be set to X'1'. SMF processes the record for subtype selectivity when both conditions are met. This support is limited to SMF record types 0 through 127 (user records are not supported).

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks publications

For information about ordering these publications, see “How to get IBM Redbooks publications” on page 252. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *OS/390 MVS Parallel Sysplex Capacity Planning*, SG24-4680
- ▶ *z/OS Workload Manager Implementation and Exploitation*, SG24-5326
- ▶ *z/OS Intelligent Resource Director*, SG24-5952
- ▶ *VSAM Demystified*, SG24-6105
- ▶ *Systems Programmer's Guide to: z/OS System Logger*, SG24-6898
- ▶ *A Solution to the Multiserver Priority Queuing Model*, REDP-3969

Other publications

These publications are also relevant as further information sources:

- ▶ *Large Systems Performance Reference*, SC28-1187
- ▶ *z/OS MVS Initialization and Tuning Guide*, SA22-7591
- ▶ *z/OS MVS Planning: Workload Management*, SA22-7602
- ▶ *z/OS MVS Programming: Workload Manager Services*, SA22-7619
- ▶ *z/OS System Management Facilities (SMF)*, SA22-7630
- ▶ *z/OS Resource Measurement Facility (RMF) User's Guide*, SC33-7990
- ▶ *z/OS Resource Measurement Facility (RMF) Report Analysis*, SC33-7991
- ▶ *z/OS MVS Programming: Authorized Assembler Services Guide*, SA22-7608

Online resources

These web sites and URLs are also relevant as further information sources.

- ▶ RMF home page
<http://www.ibm.com/servers/eserver/zseries/zos/rmf>
- ▶ *Large Systems Performance Reference for IBM zSeries and S/390* home page
<http://www-1.ibm.com/servers/eserver/zseries/lspr/zSeries.html>
- ▶ LSPR ratios are based on the workload mix that the installation is running; that methodology is described at the following sites:
<http://www-1.ibm.com/servers/eserver/zseries/lspr/lspmixmap.html>

<http://www-03.ibm.com/support/techdocs/atmastr.nsf/WebIndex/PRS135>

- ▶ SRM home page
<http://www.ibm.com/servers/eserver/zseries/srm/>
- ▶ WLM home page
<http://www.ibm.com/servers/eserver/zseries/zOS/wlm>
- ▶ Technical documents
<http://www.ibm.com/support/techdocs>
- ▶ IBM clients can obtain zPCR from the Internet at the following site:
<http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/PRS1381>
- ▶ *zCP3000 User's Guide*, Document ID PRS1772
<http://www.ibm.com/support/techdocs>
- ▶ Acrobat Reader for Windows (available for no cost)
<http://www.adobe.com>
- ▶ SoftCap is downloadable from the following sites:
 - Clients
<http://www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/PRS268>
 - Business Partners
<http://www.ibm.com/partnerworld/sales/systems>

How to get IBM Redbooks publications

You can search for, view, or download Redbooks publications, Redpapers, publications, Technotes, draft publications and Additional materials, and order hardcopy Redbooks publications, at this web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services



Redbooks

ABCS of z/OS System Programming Volume 11

(0.5" spine)
0.475" x 0.873"
250 <-> 459 pages



ABCs of z/OS System Programming

Volume 11



Capacity planning

Performance management

RMF, SMF

The ABCs of z/OS System Programming is a thirteen-volume collection that provides an introduction to the z/OS operating system and the hardware architecture. Whether you are a beginner or an experienced system programmer, the ABCs collection provides the information you need to start your research into z/OS and related subjects. If you want to become more familiar with z/OS in your current environment, or if you are evaluating platforms to consolidate your e-business applications, the ABCs collection will serve as a powerful technical tool.

The volumes contain the following content:

Volume 1: Introduction to z/OS and storage concepts, TSO/E, ISPF, JCL, SDSF, and z/OS delivery and installation

Volume 2: z/OS implementation and daily maintenance, defining subsystems, JES2 and JES3, LPA, LNKST, authorized libraries, Language Environment, and SMP/E

Volume 3: Introduction to DFSMS, data set basics, storage management hardware and software, VSAM, System-Managed Storage, catalogs, and DFSMSStvs

Volume 4: Communication Server, TCP/IP and VTAM

Volume 5: Base and Parallel Sysplex, System Logger, Resource Recovery Services (RRS), global resource serialization (GRS), z/OS system operations, automatic restart management (ARM), Geographically Dispersed Parallel Sysplex (GPDS), availability in the zSeries environment

Volume 6: Introduction to security, RACF, Digital certificates and PKI, Kerberos, cryptography and z990 integrated cryptography, zSeries firewall technologies, LDAP, Enterprise Identity Mapping (EIM), and firewall technologies

Volume 7: Printing in a z/OS environment, Infoprint Server and Infoprint Central

Volume 8: An introduction to z/OS problem diagnosis

Volume 9: z/OS UNIX System Services

Volume 10: Introduction to z/Architecture, zSeries processor design, zSeries connectivity, LPAR concepts, HCD, and HMC

Volume 11: Capacity planning, performance management, RMF, and SMF

Volume 12: WLM

Volume 13: JES3

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6327-01

ISBN 0738434299