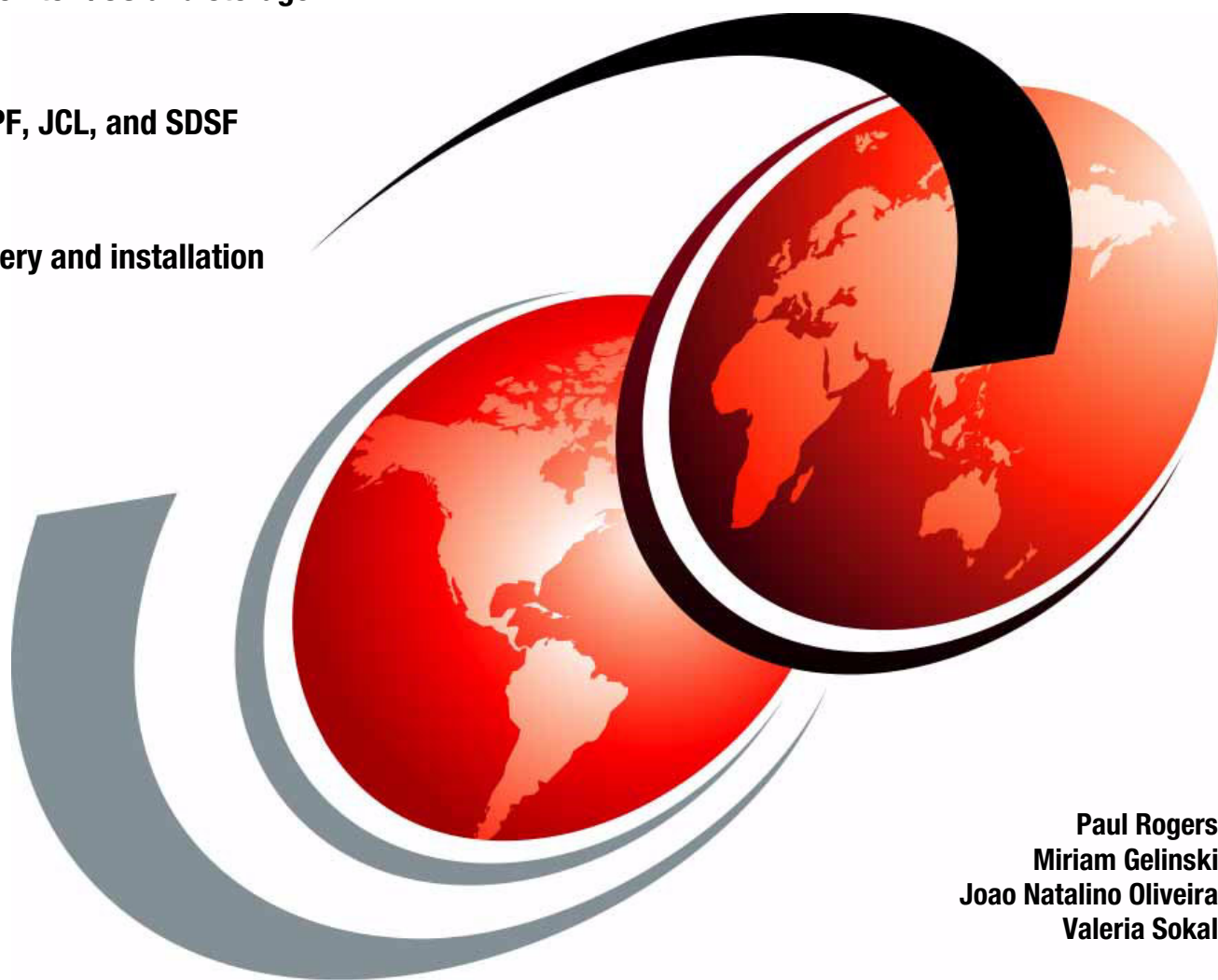


# ABCs of z/OS System Programming Volume 1

Introduction to z/OS and storage concepts

TSO/E, ISPF, JCL, and SDSF

z/OS delivery and installation



Paul Rogers  
Miriam Gelinski  
Joao Natalino Oliveira  
Valeria Sokal

**Redbooks**





International Technical Support Organization

**ABCs of z/OS System Programming  
Volume 1**

December 2003

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

**First Edition (December 2003)**

This edition applies to Version 1, Release 4, of z/OS™ (5694-A01) to Version 1, Release 4, of z/OS.e™ (5655-G52), and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright International Business Machines Corporation 2003. 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

<b>Notices</b> .....	vii
Trademarks .....	viii
<b>Preface</b> .....	ix
The team that wrote this redbook .....	ix
Become a published author .....	x
Comments welcome .....	x
<b>Chapter 1. Introduction to z/OS</b> .....	1
1.1 z/OS services .....	2
1.2 IBM server and operating system evolution .....	4
1.3 z/OS and z/OS.e .....	5
1.4 Hardware requirements .....	6
1.5 z/Series server: Basic and LPAR mode .....	7
1.6 z/OS Base Control Program (BCP) .....	8
1.7 z/OS base elements .....	9
1.8 z/OS optional features .....	10
1.9 z/OS features and elements description .....	11
1.10 z/OS Security Server: RACF component .....	16
1.11 Data Facility Storage Management Subsystem (DFSMS) .....	17
1.12 DFSMS Extended Remote Copy: XRC .....	19
1.13 z/OS Communications Server: TCP/IP .....	20
1.14 System Modification Program Extended (SMP/E) .....	21
1.15 Infoprint Server .....	22
1.16 Resource Management Facility (RMF) .....	23
1.17 System Management Facility (SMF) .....	24
1.18 Time Sharing Option/Extended (TSO/E) .....	25
1.19 UNIX System Services .....	26
1.20 Some z/OS services requiring customization .....	27
1.21 Library Lookaside (LLA) .....	28
1.22 Virtual Lookaside Facility (VLF) .....	29
1.23 Workload Manager (WLM) .....	30
1.24 z/OS operating system: the role of a system programmer .....	31
1.25 z/OS system programmer management overview .....	32
1.26 System programmer and z/OS operations .....	34
1.27 Requirements for installation .....	37
1.28 Choosing an install package .....	39
1.29 Ordering z/OS .....	41
1.30 Installing z/OS: ServerPac .....	42
1.31 Installing z/OS: Custom-built product delivery option .....	43
1.32 Installing z/OS: Fee-based packages .....	44
<b>Chapter 2. z/OS storage concepts</b> .....	45
2.1 Processor storage overview .....	46
2.2 Virtual storage concepts .....	48
2.3 Frames, slots, and pages .....	49
2.4 The address space concept .....	50
2.5 Addressing mode and residence mode .....	51
2.6 Storage managers .....	52

2.7	Virtual storage manager	53
2.8	Real storage manager	54
2.9	Auxiliary storage manager	55
2.10	Paging and swapping	56
2.11	Auxiliary data sets	57
2.12	31-bits address space map	58
2.13	The common virtual storage area	59
2.14	z/OS nucleus	60
2.15	System queue area (SQA/ESQA)	61
2.16	Common service area (CSA and Extended CSA)	62
2.17	Link pack area (LPA and Extended LPA)	63
2.18	31-bit private area	65
2.19	Data spaces and hiperspace	66
2.20	64-bit address space map	68
2.21	Size and number notation	70
2.22	Segment tables and page tables	71
2.23	31-bit virtual address	72
2.24	64-bit virtual address translation	73
2.25	Translating a 64-bit virtual address	74
2.26	System initialization	75
2.27	z/OS address spaces	76
2.28	Subsystem definitions	77
2.29	Multiprogramming and multiprocessing	79
2.30	Program compile, link-edit, and execution	80
<b>Chapter 3. TSO/E, ISPF, JCL, and SDSF</b>		<b>81</b>
3.1	z/OS facilities for system programmers	82
3.2	TSO/E	83
3.3	TSO/E highlights	84
3.4	TSO/E customization	87
3.5	TSO/E: TCAS start procedure	89
3.6	TSO/E logon procedure	90
3.7	TSO/E logon process in a VTAM environment	92
3.8	TSO/E full-screen logon panel	93
3.9	TSO/E line-mode	94
3.10	Using TSO/E as batch job	95
3.11	TSO/E <b>PROFILE</b> command	96
3.12	TSO/E languages	97
3.13	Interactive System Productivity Facility (ISPF)	98
3.14	ISPF: Data set types supported	99
3.15	ISPF: Data set and member naming conventions	100
3.16	ISPF components	101
3.17	Sample CLIST to allocate ISPF and SDSF data sets	103
3.18	ISPF primary option menu	105
3.19	ISPF panel areas	106
3.20	Action bars	107
3.21	Customizing your TSO/ISPF/PDF session	108
3.22	Allocating data sets: Utility option	109
3.23	Utility Selection Panel	110
3.24	Data Set Utility: Allocating a data set	111
3.25	Allocate New Data Set panel	113
3.26	Edit function: Option 2	115
3.27	Edit Entry Panel	116

3.28	Editing a data set . . . . .	117
3.29	ISPF edit: Some line commands . . . . .	118
3.30	ISPF edit panel: Inserting lines . . . . .	119
3.31	ISPF edit: Repeating and deleting lines . . . . .	120
3.32	Edit: Copying lines . . . . .	121
3.33	ISPF/PDF edit: Primary commands . . . . .	122
3.34	ISPF/PDF edit: Profile command . . . . .	123
3.35	ISPF/PDF edit: Saving new or updated files . . . . .	124
3.36	ISPF Data Set List Utility option . . . . .	125
3.37	Working with a data set list . . . . .	126
3.38	Data Set List Actions . . . . .	127
3.39	Job control language (JCL) . . . . .	128
3.40	JCL introduction . . . . .	129
3.41	JCL-related actions . . . . .	131
3.42	Required control statements . . . . .	132
3.43	JCL streams and jobs . . . . .	133
3.44	JES control statements in JCL . . . . .	135
3.45	Introduction to JCL: Creating a data set . . . . .	136
3.46	JCL: JOB statement . . . . .	137
3.47	JCL: EXEC statement . . . . .	139
3.48	JCL: DD statement . . . . .	141
3.49	DD statement parameters: DISP, UNIT . . . . .	142
3.50	DD statement parameters: SPACE, LRECL, BLKSIZE . . . . .	144
3.51	Submitting a job . . . . .	146
3.52	Spool Display and Search Facility (SDSF) . . . . .	147
3.53	SDSF: Panels hierarchy . . . . .	148
3.54	SDSF: Primary option menu . . . . .	149
3.55	SDSF: Options menu . . . . .	150
3.56	SDSF: Viewing the JES2 output files . . . . .	151
3.57	SDSF: Display Active Users (DA) . . . . .	153
3.58	Issuing MVS and JES commands . . . . .	154
3.59	SDSF: Input queue panel . . . . .	155
3.60	SDSF: Output queue panel . . . . .	156
3.61	SDSF: Held Output queue panel . . . . .	157
3.62	SDSF: Status panel . . . . .	158
3.63	SDSF: Tutorial . . . . .	159
<b>Chapter 4. MVS delivery and installation . . . . .</b>		<b>161</b>
4.1	z/OS installation overview . . . . .	162
4.2	z/OS release cycle . . . . .	163
4.3	z/OS delivery options . . . . .	164
4.4	ServerPac service level . . . . .	166
4.5	CBPDO service level . . . . .	167
4.6	System and installation requirements . . . . .	168
4.7	Reviewing your current system . . . . .	169
4.8	The driving and target system . . . . .	170
4.9	z/OS installation using ServerPac . . . . .	171
4.10	Installing the CustomPac dialogs . . . . .	172
4.11	The RIM tape samples . . . . .	173
4.12	Starting the CustomPac dialogs . . . . .	174
4.13	Receiving the ServerPac order . . . . .	175
4.14	Order Receive panel . . . . .	176
4.15	Generate Jobstream panel . . . . .	177

4.16	Selecting an order to Install	178
4.17	Installation dialog	179
4.18	Choosing the installation type	180
4.19	Selecting a JES for the configuration	181
4.20	Create Configuration panel	182
4.21	Configuration Profile panel	183
4.22	Installation Variables panel	184
4.23	Define ZONE Information panel	185
4.24	Modify System Layout Options panel	186
4.25	Summary of features and elements	187
4.26	Summary of data sets of a feature/element	188
4.27	Summary of Physical Volumes panel	189
4.28	Creating the recommended system layout	190
4.29	Current Volume Configuration panel	191
4.30	Display and Change Volume Attributes panel	192
4.31	Define alias-to-catalog relationships	193
4.32	Define system-specific alias (SSA)	194
4.33	Define SSA and CATALOG Data panel	195
4.34	Job Selection List panel	196
4.35	GENERATE File Tailored Installation Jobs	197
4.36	Generate installation jobs	198
4.37	Displaying the processing log	199
4.38	Save configuration panel	200
	<b>Related publications</b>	201
	IBM Redbooks	201
	Other publications	201
	Online resources	201
	How to get IBM Redbooks	202



# Notices

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

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

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

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

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

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

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

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

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


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

## COPYRIGHT LICENSE:

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

# Trademarks

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

@server™  
@server™  
Redbooks (logo) ™  
ibm.com®  
z/Architecture™  
z/OS®  
zSeries®  
AIX®  
BookManager®  
CICS®  
CUA®  
Domino®  
DB2®  
DFS™  
DFSMS/MVS®  
DFSMSdftp™  
DFSMSdss™  
DFSMSshsm™  
DFSMSrmm™  
DFSORT™  
Encina®  
ES/9000®

ESCON®  
First Failure Support Technology™  
FlashCopy®  
FFST™  
GDDM®  
Hiperspace™  
Infoprint®  
Intelligent Miner™  
IBM®  
IBMLink™  
IMS™  
IMS/ESA®  
Language Environment®  
Lotus®  
MVS™  
MVS/ESA™  
MVS/SP™  
NetView®  
Open Class®  
OpenEdition®  
OS/2®  
OS/390®

Parallel Sysplex®  
Print Services Facility™  
ProductPac®  
PR/SM™  
Redbooks™  
RACF®  
RETAIN®  
RMF™  
S/370™  
S/390®  
ServicePac®  
System/370™  
SystemPac®  
SNAP/SHOT®  
SOM®  
TotalStorage®  
VTAM®  
WebSphere®  
z/Architecture™  
z/OS®  
z/VM®  
zSeries®

The following terms are trademarks of other companies:

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

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

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

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

# Preface

The ABCs of z/OS® System Programming is a ten 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 that you need to start your research into z/OS and related subjects. If you would like 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.

Volume 1 provides an understanding of the software and zSeries® architecture and how it is used together with the z/OS operating system. This includes the main components of z/OS needed to customize and install the z/OS operating system.

Specific topics covered in this volume are:

- ▶ Introduction to the products and components that make up a z/OS system
- ▶ z/OS storage concepts
- ▶ The main components needed to install and customize z/OS: TSO/E, ISPF, JCL, and SDSF
- ▶ The z/OS delivery options and the download process using the ServerPac option

The contents of the other volumes are as follows:

Volume 2: z/OS implementation and daily maintenance, defining subsystems, JES2 and JES3, LPA, LNKLST, authorized libraries, catalogs

Volume 3: Introduction to DFSMS, storage management

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

Volume 5: Base and Parallel Sysplex®, system logger, global resource serialization, z/OS system operations, automatic restart management, hardware management console, performance management

Volume 6: RACF®, PKI, LDAP, cryptography, Kerberos, and firewall technologies

Volume 7: Infoprint® Server, Language Environment®, and SMP/E

Volume 8: z/OS problem diagnosis

Volume 9: z/OS UNIX System Services

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

## The team that wrote this redbook

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

**Paul Rogers** is a Consulting IT Specialist at the International Technical Support Organization, Poughkeepsie Center. He writes extensively and teaches IBM classes

worldwide on various aspects of z/OS and OS/390®. Before joining the ITSO 15 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. He has worked for IBM for 36 years.

**Miriam Gelinski** is a staff member of Maffei Consulting Group, where she is responsible for supporting customer planning and installing zSeries software. Miriam holds a bachelor degree in Information Systems from Universidade São Marcos. Before joining Maffei Consulting Group, she worked for IBM zSeries brand for two years, where she was responsible for implementing new workloads on zSeries.

**Joao Natalino Oliveira** is a certified I/T consulting specialist for IBM zSeries in Brazil, where he provides support for Brazil and Latin America. He has 28 years of experience in large systems, including MVS™, OS/390, and z/OS. His areas of expertise include performance and capacity planning, server consolidation and system programming. He holds a bachelor degree in Mathematics and Data Processing from Fund. Santo Andre, Brazil.

**Valeria Sokal** is an MVS system programmer at an IBM customer. She has 14 years of experience as a mainframe system programmer.

## Become a published author

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

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

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

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

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

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

[ibm.com/redbooks](http://ibm.com/redbooks)

- ▶ Send your comments in an Internet note to:

[redbook@us.ibm.com](mailto:redbook@us.ibm.com)

- ▶ Mail your comments to:

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



# Introduction to z/OS

z/OS is an integrated enterprise server operating system. It incorporates into one product a leading-edge and open communications server, distributed data and file services, Parallel Sysplex system support, object-oriented programming, distributed computer environment (DCE), and an open application interface. As such, it is uniquely suited to integrate today's heterogeneous and multi-vendor environments.

By incorporating the base operating system, it continues to build on the classic strengths of MVS: its reliability, continuous availability features, and security. This provides a scalable system that supports massive transaction volumes and large numbers of users with high performance, as well as advanced system and network management, security, and 24/7 availability.

This chapter presents an overview of the z/OS operating system:

- ▶ The z/OS operation system evolution
- ▶ The hardware required to install and run z/OS
- ▶ Differences between z/OS and z/OS.e
- ▶ The z/OS base and optional products, with a brief description of some of them
- ▶ The z/OS products requiring customization
- ▶ The system programmer skills needed to install and maintain the z/OS operating system
- ▶ The requirements to install z/OS
- ▶ The installation package delivery options
- ▶ z/OS 1.4 highlights

## 1.1 z/OS services

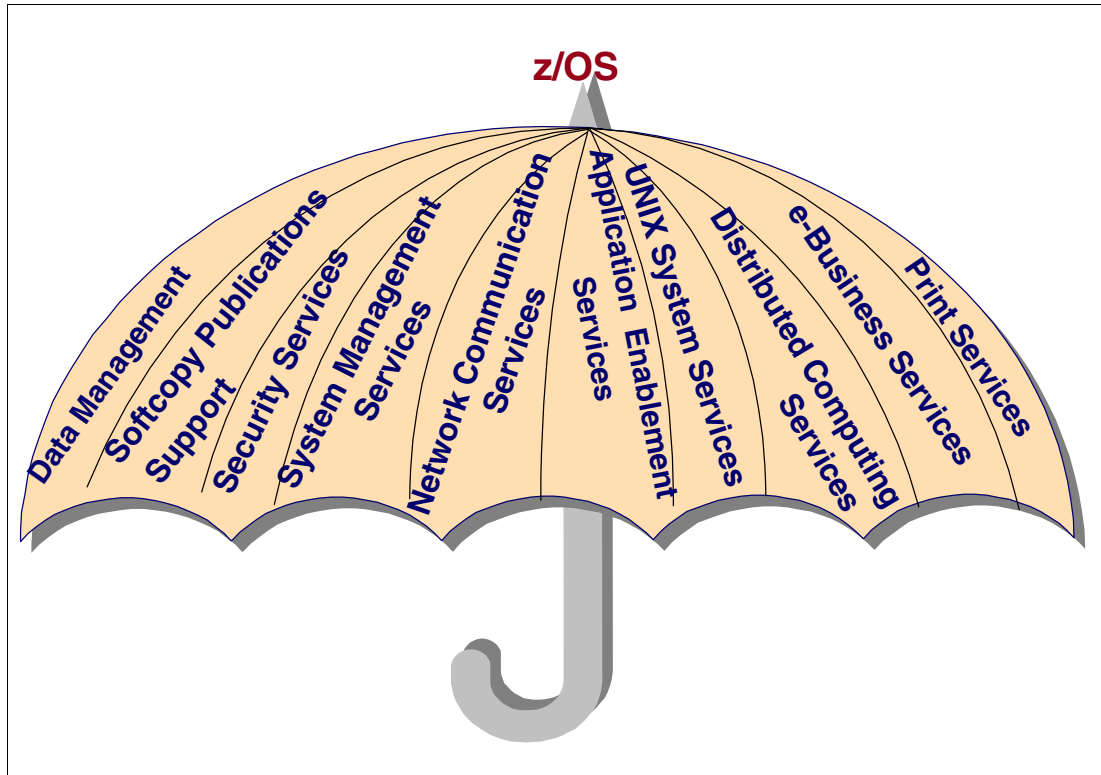


Figure 1-1 z/OS services

The z/OS system provides solutions for the following major areas:

- ▶ **Data management:** z/OS provides a set of functions to manage storage resources on the system, support storage and retrieval of data on disk, optical and tape devices, program management functions, and device management functions to define and control the operation of input and output storage devices. Distributed File Manager (DFM) supports access to remote data and storage resources.
- ▶ **Softcopy publications services:** Improve productivity in systems installation and management.
- ▶ **Security services:** Security and Cryptographic Services are a set of products and features used to control access to resources, and to audit and manage the accesses with appropriate centralized or decentralized control. These services form the basis for all security services for traditional applications, UNIX applications, and distributed systems.
- ▶ **System management services:** The functions and features provided with z/OS allow robust control and automation of the basic processes of z/OS, increasing availability, improving productivity of system programmers, and providing a consistent approach for configuring z/OS components of products.
- ▶ **Network communication services:** z/OS enables world class TCP/IP and SNA networking support; multivendor, multiplatform connectivity; connectivity to a broad set of users; and support for multiple protocols.
- ▶ **Applications enablement services:** Provide solid infrastructure in which one can build new applications, extend existing applications, and run OLTP and batch processes.
- ▶ **UNIX System Services:** z/OS contains the UNIX applications services (shell, utilities, and debugger) and the UNIX system services (kernel and runtime environment). The shell and

utilities provide the standard command interface familiar to interactive UNIX users. z/OS includes all the commands and utilities specified in the X/OPEN XPG4.2. With Language Environment, z/OS supports industry standards for C programming, shell and utilities, client/server applications, and the majority of the standards for thread management, allowing transparent data exchange and easy portability of applications in an open environment.

- ▶ **Distributed computing services:** The distributed computing services are achieved by a set of features and functions. Network File System acts as a file server to workstations, personal computers, or other authorized systems in a TCP/IP network. Remote files are mounted from the mainframe (z/OS) to appear as local directories and files on the client system. DCE enables data encryption standard (DES) algorithms and the commercial data masking facility (CDMF). Distributed File Services (DFS™) SMB allows users to access data in a distributed environment across a wide range of IBM and non-IBM platforms. SMB can automatically handle the conversion between ASCII and EBCDIC.
- ▶ **e-Business services:** The IBM HTTP Server provides for scalable, high performance Web serving for critical e-business applications. It is exclusive to z/OS. This element was previously known as a base element of z/OS under the names Lotus® Domino® Go, the Internet Connection Secure Server (ICSS) and the Internet Connection Server (ICS).
- ▶ **Print services:** Application output can be electronically distributed and printed or presented over the Web.

z/OS Release 1 was introduced as a replacement for OS/390. There were ten releases of OS/390, with a new release shipped every six months. This book is based on the current z/OS Release 1 Version 4. This Release was made generally available in September 2002.

z/OS (program number 5694-A01), the next generation of the OS/390 operating system, enables you to manage the volatility of e-business workloads. z/OS delivers the highest qualities of service for enterprise transactions and data, and extends these qualities to new applications using the latest software technologies. Some highlights of z/OS are:

- ▶ The 64-bit z/Architecture implemented by z/OS and the IBM @server zSeries server eliminates bottlenecks associated with the lack of addressable memory. The 64-bit real (central) storage support eliminates expanded storage, helps eliminate paging, and may allow you to consolidate your current systems into fewer logical partitions (LPARs) or to a single native image.
- ▶ The Intelligent Resource Director (IRD) expands the capabilities of the workload manager (WLM) by enabling resources to be dynamically managed across LPARs based on workload priorities.

## 1.2 IBM server and operating system evolution

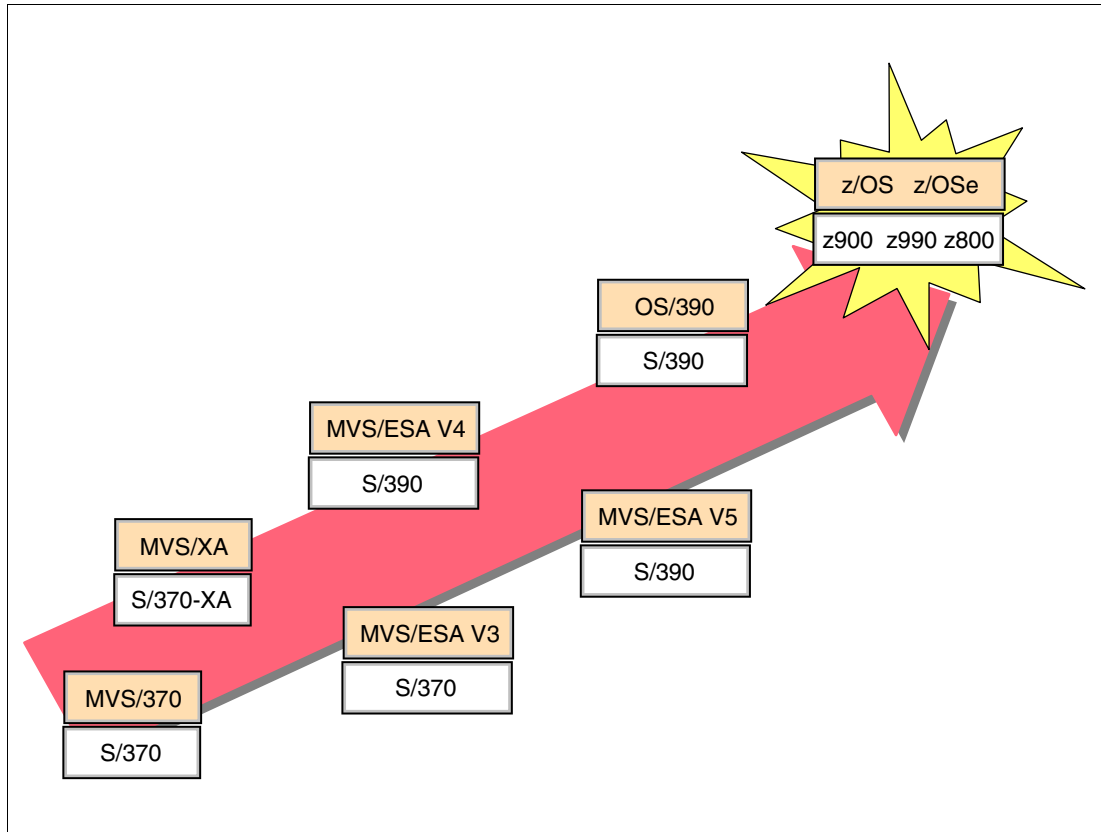


Figure 1-2 IBM server and operating system evolution

Enterprise system hardware was transformed by the introduction of the z/OS Parallel Servers. Based on the complementary metal oxide semiconductor (CMOS) technology, these parallel systems are smaller in size and larger in capacity than their predecessors, and deliver performance at a lower cost.

With the introduction of the S/390® Parallel Servers in 1994, the mainframe was revived. The CMOS technology offers mainframe computing power at a lower cost. The zSeries CMOS machines can be connected with other zSeries or S/390 CMOS machines or even traditional ES/9000® machines to form a Parallel Sysplex. The sysplex offers high availability and the option to add capacity in small increments.

The transformation of mainframe hardware together with business requirements for information technology (IT) are the driving forces behind changes in the operating system software. In 2001, the z/OS system was introduced as the zSeries server operating system. z/OS extends S/390's architecture to provide the enterprise-wide client/server infrastructure and tools that businesses need for fast, flexible deployment of new applications.



## 1.3 z/OS and z/OS.e

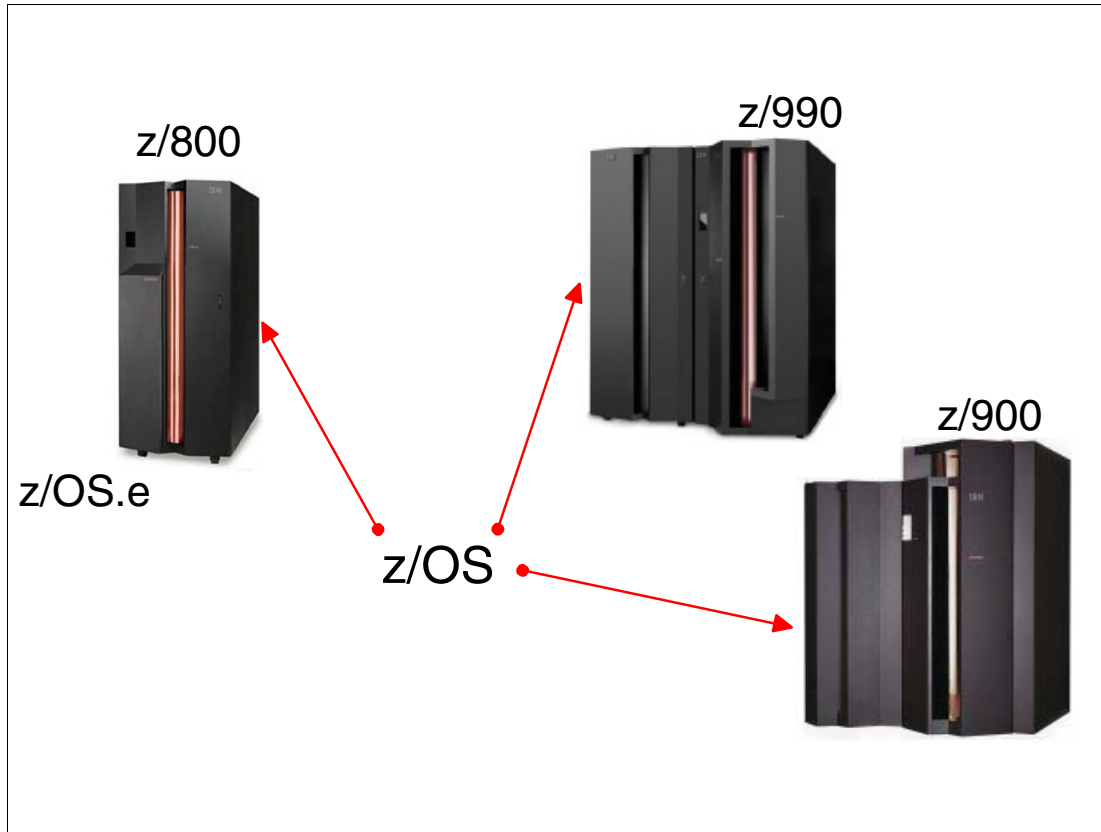


Figure 1-3 z/OS and z/OS.e Operating System

z/OS, the next generation of the OS/390 operating system, enables you to manage the volatility of e-business workloads. z/OS delivers the highest qualities of service for enterprise transactions and data, and extends these qualities to new applications using the latest software technologies.

z/OS.e is a lower-priced version of z/OS and is designed to handle new e-business workloads like:

- ▶ WebSphere® Application Server, IBM Developer Kit for the Java™ Platform (includes J2EE), DB2® V7, V6, V5, C/C++
- ▶ WebSphere Commerce (WC), DB2 database serving, Domino, and financial applications

z/OS.e offers all of the qualities of service of z/OS (availability, scalability, reliability, security) but lacks some of the functions that support traditional workloads. z/OS.e is able to run applications written exclusively in Java, Enterprise Java (J2EE) and C/C++.

z/OS can run in any IBM zSeries Server (z900, z800, z990). The only IBM server on which z/OS.e runs is in the zSeries 800 (z800) server or in any comparable non-IBM server.

z/OS and z/OS.e use the same operating system software (their code is identical). Upon Initialization Programming Load (IPL), custom parameters invoke an operating environment that is comparable to z/OS in all aspects of operation, service, management, reporting, and zSeries hardware functionality. No new skills are required for z/OS.e.

## 1.4 Hardware requirements

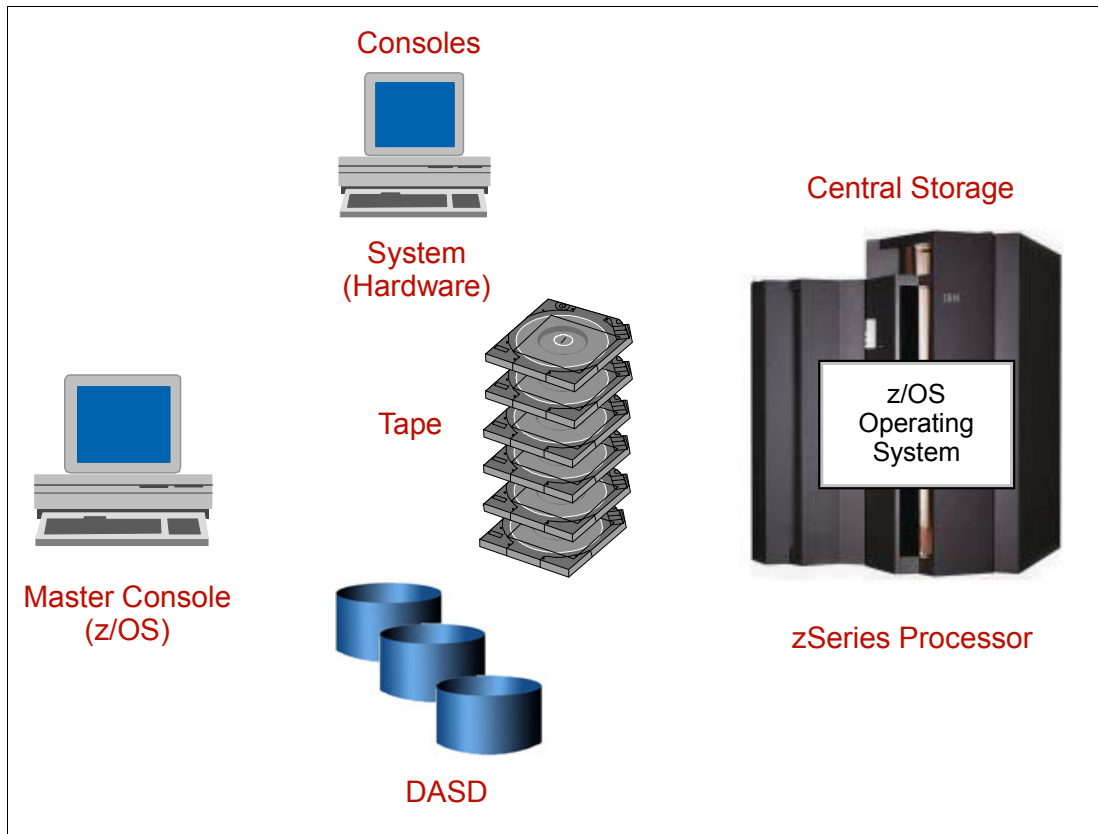


Figure 1-4 Hardware requirements

The base z/OS operating system executes in a processor and resides in the processor storage during execution. The z/OS operating system is commonly referred to as the system software.

The hardware consists of the processors and other devices such as a direct access storage device (DASD), tape, and consoles. Tape and DASD are used for system functions and by user programs that execute in a z/OS environment. When you order z/OS, you receive your order on tape cartridges. When you install the system from tape, the system code is then stored on DASD volumes. Once the system is customized and ready for operation, system consoles are required to start and operate the z/OS system. Not shown in Figure 1-4 are the control units that connect the CPU (processor) to the other tape, DASD, and console devices. The main concepts shown here are:

- Software** The z/OS operating system consists of load modules and is often called executable code. These load modules are placed onto DASD volumes in load libraries during a system install process.
- Hardware** The system hardware consists of all the devices, controllers, and processors that make up a z/OS complex.
- Devices** Shown in the figure are the tape, DASD, and console devices. There are many other types of devices that are discussed later in this document.
- Storage** Central storage, often called real or main storage, is where the z/OS operating system executes. Also, all user programs share the storage of the processor with the operating system.

## 1.5 z/Series server: Basic and LPAR mode

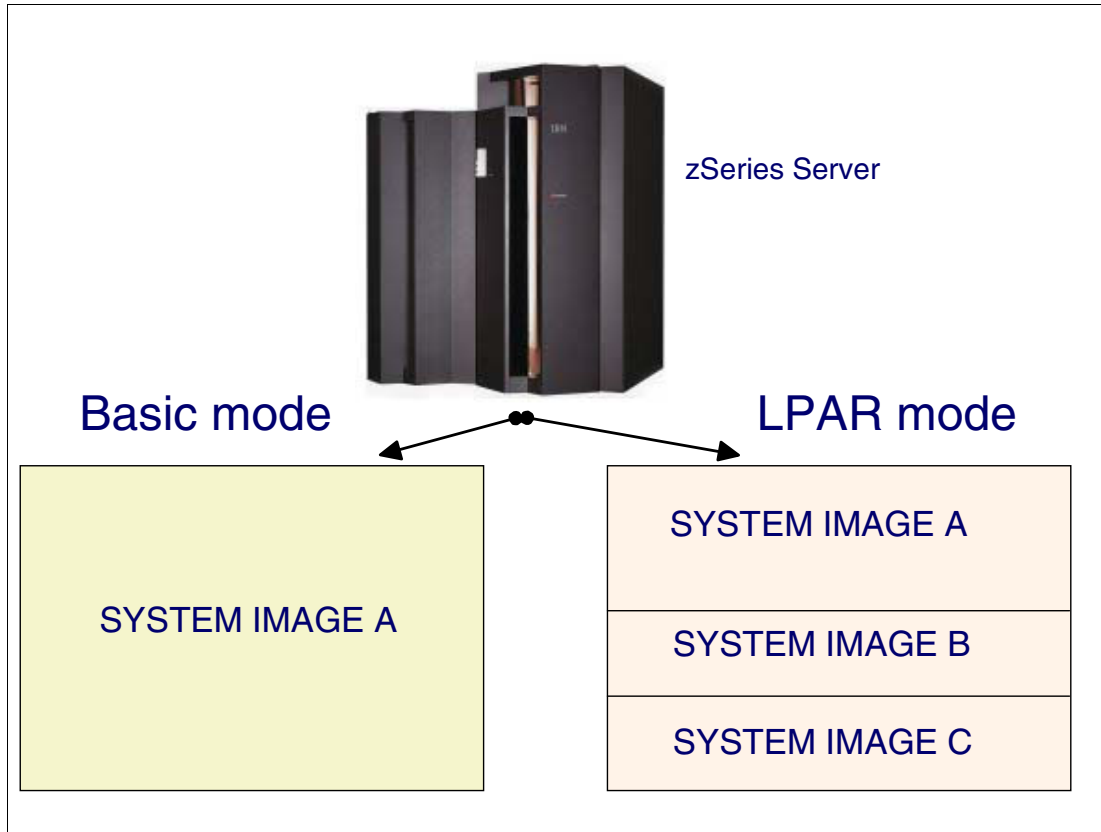


Figure 1-5 CPC: Basic and LPAR mode

The hardware can be used in two modes:

**LPAR mode** Logically partitioned (LPAR) mode is a central processor complex (CPC) power-on reset mode that enables use of the Processor Resource/System Manager (PR/SM™) feature and allows an operator to allocate CPC hardware resources (including central processor, central storage, and channel paths) among logical partitions. z/OS as the operating system runs in each LPAR in the machine.

**Basic mode** This is a central processor mode that does not use logical partitioning, with the CPC running one copy of the z/OS operating system.

## 1.6 z/OS Base Control Program (BCP)

- Essential operating system services
- I/O Configuration Program (IOCP)
- Workload Manager (WLM)
- z/OS UNIX System Services (z/OS UNIX)
- z/OS UNIX kernel
- Support for unicode
- Program management

*Figure 1-6 z/OS BCP functions*

The Base Control Program (BCP) is the backbone of the z/OS system. The BCP provides the essential services that make z/OS the system of choice when you need to process your workloads reliably, securely, with complete data integrity, and without interruption.

The BCP includes the I/O configuration program (IOCP), the workload manager (WLM), system management facilities (SMF), the z/OS UNIX System Services kernel, support for Unicode, and program management functions.

z/OS UNIX System Services was formerly called OpenEdition® System Services. It was a non-exclusive base element of OS/390 V2R1, an exclusive element of OS/390 V2R2, and was integrated into the BCP element in OS/390 V2R3. In OS/390 V2R6 it remained as a part of the BCP, but its name was changed to OS/390 UNIX System Services; it is now called z/OS UNIX System Services.

The Unicode Standard is the universal character encoding standard used for representation of text for computer processing. The Unicode Standard provides the capacity to encode all of the characters used for the written languages of the world. The z/OS support for Unicode implements these Unicode Version 3.0 standards: Character codepage and case conversion and normalization. The Unicode support also takes advantage of native z/Architecture Unicode HW instructions for faster processing.

## 1.7 z/OS base elements

Base Control Program (BCP)	High Level Assembler (HLASM) IBM
Bulk Data Transfer base (BDT)	HTTP Server and Text Search
BookManager BookServer	ICKDSF
BookManager Read	ISPF
C/C++ IBM Open Class Library	JES2
Cryptographic Services	Language Environment
DCE Application Support	MICR/OCR support
DCE Base Services	msys for Operations
DFSMSdfp	msys for Setup
Distributed File Service	Network File System (NFS)
Encina Toolkit Executive	OSA Support Facility
EREP	SMP/E
ESCON Director Support	TSO/E
FFST	TIOC
GDDM	z/OS Communication Server
HCD	z/OS UNIX
	3270 PC File Transfer Program

Figure 1-7 z/OS base elements

The z/OS system consists of base elements that deliver essential operating functions—in addition to the services provided by the BCP functions—such as communications support, online access, host graphics, and online viewing of publications.

The base elements are listed in Figure 1-7. When you order z/OS or z/OS.e, you receive all of the base elements. However, with z/OS.e, some base elements are not functional or not licensed for use, or both.

Shipped as part of the z/OS system are the base operating system, products, and features; for example, UNIX System Services and LAN services. In addition to these features, products such as TSO/E, ISPF, GDDM®, and BookManager® READ, which provide essential operating system functions, are included in the base and are called base elements. Some of the base elements can be dynamically enabled and disabled. The reason for this is that a customer may choose to use a vendor product instead of IBM products.

The idea of the z/OS system is to have elements and features instead of program products. This concept might be more easily explained by saying that z/OS consists of a collection of functions that are called base elements and optional elements. The optional elements (features) are either integrated or nonintegrated. It is important to note that these optional features, both integrated and nonintegrated, are also tested as part of the integration of the entire system.

## 1.8 z/OS optional features

BDT File-to-File	HLASM Toolkit
BDT	IBM HTTP Server NA Secure
BookManager BUILD	Infoprint Server
C/C++ with Debug Tool	JES3
C/C++ w/o Debug Tool	Network Authentication Service Level 3
DFSMSdss	NJE
DFSMSHsm	Open Cryptographic Facility Security Level 3
DFSMSrmm	RMF
DFSMSStvs	SDSF
DFSORT	Security Server
GDDM-PGF	SNA
GDDM-REXX	System SSL Security Level 3
HCM	z/OS Communications Server Security Level 3

Figure 1-8 z/OS optional features

In addition to the base elements, z/OS has optional features that are closely related to the base features. The optional features are orderable with z/OS or z/OS.e and provide additional operating system functions. The optional features are listed in Figure 1-8. Some optional features that are orderable with z/OS are not orderable with z/OS.e.

Optional features are unpriced or priced:

- ▶ *Unpriced* features are shipped to you only if you order them.
- ▶ *Priced* features are always shipped. These features are ready to use after you install z/OS or z/OS.e (and customize them as needed). IBM enables the priced features you ordered and disables the priced features you did not order. Later on, if you decide to use them, you can notify IBM, and then you enable them dynamically (which is known as dynamic enablement). Dynamic enablement is done by updating SYS1.PARMLIB member IFAPRDxx; notify IBM by contacting your IBM representative.

Some *optional* features that support dynamic enablement are always shipped. Examples are JES3, DFSMSdss™, and DFSMSHsm™. If these features are ordered as part of the z/OS system order, they are shipped as enabled in the system. If they are not ordered, they are shipped as disabled. Later on you can enable them through a SYS1.PARMLIB member.

The other type of features are the optional features equivalent to optional program products. Examples are RACF from the Security Server set of programs, RMF™, C/C++ compiler, and so on.

## 1.9 z/OS features and elements description

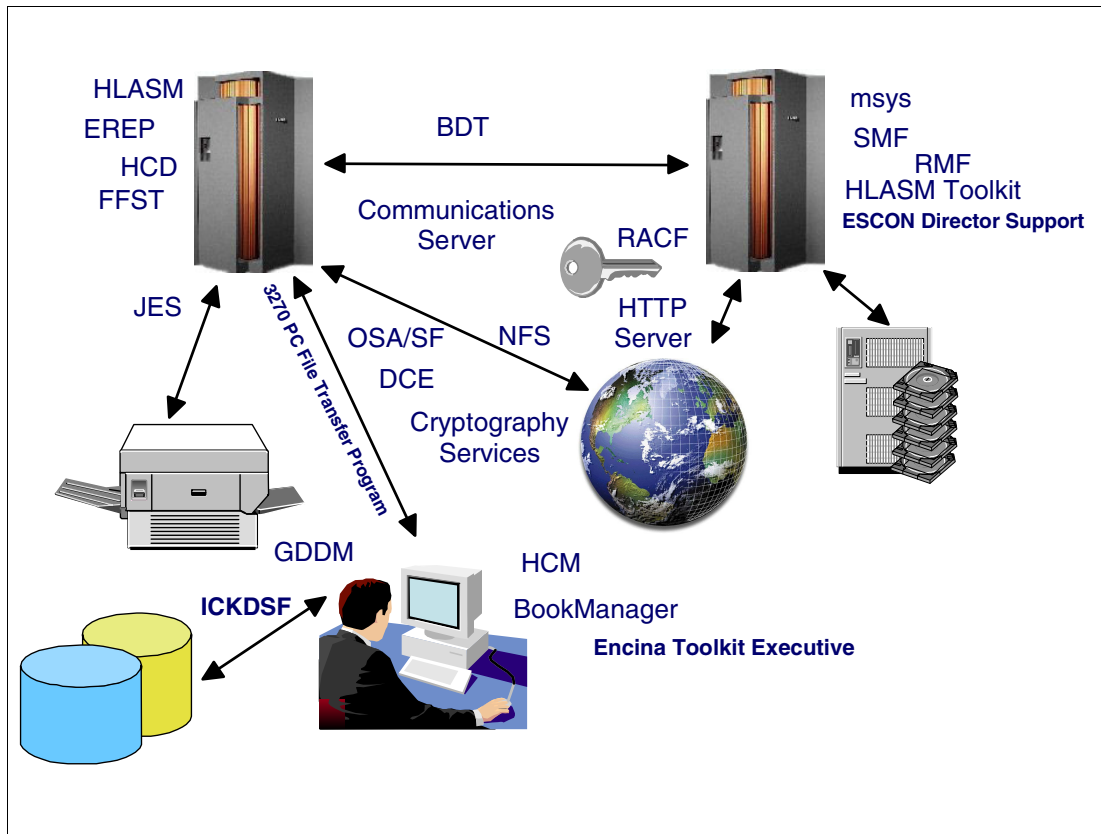


Figure 1-9 z/OS features and elements

The z/OS elements and features list is something like a soup of letters. Here we give you a brief introduction to some of them; in the following sections, we present more details about the rest.

- ▶ **Bulk Data Transfer (BDT):** An exclusive, base element that provides the base services that the optional BDT features need to transfer data from one computer system to another. The optional, exclusive features are:
  - **BDT File-to-File:** Allows users at one z/OS system in an SNA network to copy data sets to or from another z/OS system in the network.
  - **BDT SNA NJE:** Allows JES3 users to transmit jobs, output, commands, and messages from one computer system to another within an SNA network. This feature is related to the feature JES3.
- ▶ **BookManager BookServer:** An exclusive, base element that converts BookManager documents to HTML for display through a Web browser.
- ▶ **BookManager BUILD:** An optional feature that creates softcopy documents that can be used by any of the BookManager products, such as BookManager READ or BookManager BookServer.
- ▶ **BookManager READ:** A base element used to display, search, and manage online documents and bookshelves. A related optional feature is BookManager BUILD.
- ▶ **C/C++ IBM Open Class® Library:** An exclusive, base element that provides a set of C++ class libraries. C/C++ applications can run without requiring a license for either the C/C++

with Debug Tool or C/C++ without Debug Tool features. Also, applications can access the required dynamic link libraries (DLLs).

- ▶ **Communications Server:** An exclusive, base element that supports secure TCP/IP, SNA, and UNIX networking throughout an enterprise. It gives you the ability to connect subsystems and applications to each other, and to connect network devices (such as terminals and printers) to the system. Communications Server consists of two components: IP Services and SNA Services.
- ▶ **Communications Server Security Level 3:** This exclusive, optional feature works in conjunction with the Communications Server base element to provide stronger encryption (greater than 64 bits) than that available without this feature. This feature uses the TDES algorithm for encryption. The actual level of encryption that takes place with this feature installed can be configured to be something less than the maximum level enabled by the feature. This feature is related to the base element Communications Server and to the firewall technologies component of the Security Server feature.
- ▶ **Cryptography Services:** Cryptography is the transformation of data to conceal its meaning. Cryptography Services is an exclusive, base element that provides the following base cryptographic functions: data secrecy, data integrity, personal identification, digital signatures, and the management of cryptographic keys. Keys as long as 56 bits are supported by this base element. Keys longer than 56 bits are supported by the related optional features OCSF Security Level 3 and System SSL Security Level 3.
- ▶ **DCE Application Support:** An exclusive, base element that facilitates the interaction between Distributed Computing Environment (DCE) clients and CICS® or IMS™ regions.
- ▶ **DCE Base Services:** An exclusive, base element that provides services for developing and running client/server applications, including remote procedure call, directory, security, and distributed time services. DCE Base Services uses the limited DES algorithm for encryption. This element is at the Open Group Open Software Foundation (OSF) DCE 1.1 level.
- ▶ **DFSORT™:** An optional feature that sorts, merges, and copies data.
- ▶ **Distributed File Service:** An exclusive, base element that provides:
  - The DCE file serving (DFS(TM)) component of the Open Group Open Software Foundation (OSF) DCE. The file serving support (the DFS client and server) is at the OSF 1.2.2 level.
  - The zSeries File System (zFS). The zFS is a UNIX file system that can be used in addition to the Hierarchical File System (HFS). zFS file systems contain files and directories that can be accessed with the z/OS and z/OS.e hierarchical file system file APIs. The zFS provides significant performance gains in most environments requiring files 8 KB in size or greater that are frequently accessed and updated. The access performance of smaller files is equivalent to the HFS. For all files, the zFS provides a reduced exposure to loss of updates. The zFS is a logging file system with a write pattern to disk that reduces the points of failure after a system outage.
  - Server message block (SMB) file/print serving support. DFS includes an SMB function that is based on the X/Open SMB Version 2 specification and the IETF RFCs on Netbios over IP. Included in the support is access to HFS, sequential, PDS, PDSE, and VSAM data sets from Windows® 98, Windows NT® 4.0, and Windows 2000 Professional workstations. Windows workstation users can also exploit z/OS and z/OS.e printer capabilities using the SMB file/print server interface to the z/OS or z/OS.e Infoprint Server feature.
- ▶ **Encina® Toolkit Executive:** An exclusive, base element that provides a set of tools for developing client components of distributed transactional applications.



- ▶ **EREP:** The Environmental Record Editing and Printing Program (EREP) is a base element that edits and prints reports for the records placed in the error recording data set (ERDS), helping IBM service representatives fix problems.
- ▶ **ESCON® Director Support:** This exclusive, base element enables the reporting of ESCON director device errors to z/OS or z/OS.e.
- ▶ **FFST™:** First Failure Support Technology™ ((FFST) is an exclusive, base element that provides immediate notification and first failure data capture for software events.
- ▶ **GDDM:** A base element that provides presentation services and device-driving capability. It includes PCLK and REXX code. Related optional features are:
  - **GDDM-Presentation Graphics Feature (PGF)** is a set of programs for creating presentation material in a variety of styles.
  - **GDDM-REXX** is a productivity tool that enables programmers to prototype GDDM applications and to create small routines and utility programs quickly and easily.
- ▶ **HCD:** Hardware Configuration Definition (HCD) is an exclusive, base element that defines both the operating system configuration and the processor hardware configuration for a system. A related optional feature is:
  - **HCM:** Hardware Configuration Manager is an exclusive, optional feature that is a client/server interface to the base element HCD.
- ▶ **HLASM:** High Level Assembler (HLASM) is a base element that integrates almost all functions of past assemblers and provides extensions and improvements. A related optional feature is:
  - **HLASM Toolkit:** Provides tools to improve application development, debugging, and recovery.
- ▶ **IBM HTTP Server:** An exclusive, base element, it is the Web server for z/OS and z/OS.e. It provides scalable, high performance Web serving for critical e-business applications. It supports Secure Sockets Layer (SSL) secure connections, dynamic caching using the Fast Response Cache Accelerator, multiple IP addresses, proxy authentication, and double-byte character set characters. A related optional feature is:
  - **IBM HTTP Server NA Secure:** Supports SSL connections using 128-bit encryption. This feature uses the System SSL component of base element Cryptographic Services for encryption.
- ▶ **ICKDSF:** Device Support Facility (ICKDSF) is a base element that enables you to perform functions needed for the installation and use of DASD.
- ▶ **ILM:** IBM License Manager is an exclusive, base element. It was planned to be a combination of license management tools that you would use to manage licenses and check compliance with software terms and conditions. This element is not delivered yet (as of z/OS V1R4).
- ▶ **JES2 or JES3:** BCP uses a job entry subsystem (JES) to receive jobs into the operating system, schedule them for processing by BCP, and control their output processing. Basically, by separating job processing into a number of tasks, z/OS operates more efficiently. In a simple view, z/OS divides the management of jobs and resources between the JES and the base control program of z/OS. JES2 manages jobs before and after running the program; the base control program manages them during processing of their output, then JES2 purges them from the system. For an installation that has more than one processor in a configuration, there are noticeable differences in how JES2 exercises independent control over its job processing functions. That is, within the configuration, each JES2 processor controls its own job input, job scheduling, and job output processing. In contrast, JES3 exercises centralized control over its processing functions through a

single global JES3 processor. JES2 is an exclusive, base element and JES3 is an exclusive, optional element.

- ▶ **Language Environment (LE):** An exclusive, base element that provides the *run-time* environment for programs generated with C, C++, COBOL, Fortran, and PL/I. LE does not replace the need for separate compilers.
- ▶ **MICR/OCR:** An exclusive, base element that provides the device support code for various magnetic and optical devices.
- ▶ **msys for Operations:** An exclusive, base element. Managed System Infrastructure for Operations (msys for Operations) simplifies the day-to-day operation of z/OS and z/OS.e Parallel Sysplex configurations by automating typical operator tasks and events.
- ▶ **msys for Setup:** An exclusive, base element. Managed System Infrastructure for Setup (msys for Setup) offers a new approach for configuring z/OS, z/OS.e, and products that run on z/OS and z/OS.e. The configuration process is driven by a graphical user interface that greatly facilitates the definition of customization parameters. Updates are under the control of the msys for Setup user and are made directly to the system.
- ▶ **NFS:** Network File System is an exclusive, base element that acts as a file server to workstations, personal computers, or other authorized systems in a TCP/IP network. It consists of a client (Network File System Client) and a server (Network File System Server). It supports Berkeley sockets but not TCP/IP sockets.
- ▶ **OCSF Security level 3:** An exclusive, optional feature that works in conjunction with the OCSF component of the Cryptographic Services base element to provide stronger encryption (greater than 64 bits) than that available without this feature. This feature uses the TDES, DES, and RC2/RC4/RC5 algorithms for encryption.
- ▶ **OSA/SF:** An exclusive, base element. Open Systems Adapter/Support Facility (OSA/SF) provides a user-friendly interface for monitoring and controlling the zSeries Open Systems Adapter feature, which provides zSeries network connectivity directly to local area networks (LANs) and wide area networks (WANs) that support IP and SNA protocols. OSA/SF supports Gigabit, Token Ring, Fast Ethernet, 1000Base-T Ethernet, and ATM features depending on the processor on which z/OS runs.
- ▶ **SDSF:** An exclusive, optional feature. System Display and Search Facility (SDSF) provides you with information to monitor, manage, and control your z/OS or z/OS.e system.
- ▶ **Security Server:** An exclusive, optional feature. Security Server lets you control access to protected resources. Security Server consists of the following components:
  - DCE Security Server, which uses the limited DES algorithm for encryption.
  - Firewall Technologies: The Internet Security Association and Key Management Protocol (ISAKMP) server and the configuration server of Firewall Technologies are packaged with the Security Server but licensed with the base operating system, and can be used without licensing or enabling the Security Server. The ISAKMP server implements the required elements of Internet Key Exchange (IKE) as defined by Request for Comments (RFC) 2409. The Configuration server communicates with the firewall configuration graphical user interface (GUI) that is shipped within Firewall Technologies. Firewall Technologies uses the DES algorithm for encryption.
  - LDAP Server (Note that LDAP Client was merged into LDAP Server.) LDAP Server is licensed with the base operating system and can be used without ordering or enabling Security Server. LDAP Server uses the System SSL component of base element Cryptographic Services for encryption.
  - Network Authentication Service is licensed with the base operating system and can be used without ordering or enabling Security Server. Network Authentication service uses the DES algorithm for encryption.

- Open Cryptographic Enhanced Plug-ins (OCEP) is licensed with the base operating system and can be used without ordering or enabling Security Server.
- RACF uses the limited DES and CDM algorithm, and the RC2 40-bit algorithm, for encryption.
- Public Key Infrastructure (PKI) Services is licensed with the base operating system and can be used without ordering or enabling Security Server. This component uses RACF, the OCSF component of base element Cryptographic Services, and the ICSF component of base element Cryptographic Services for encryption.
- ▶ **Network Authentication Security Level 3:** An exclusive, optional feature. This feature works in conjunction with the Network Authentication Service component of the Security Server feature to provide stronger encryption (greater than 64 bits) than that available without this (Level 3) feature. This feature uses the TDES algorithm for encryption.
- ▶ **System SSL Security Level 3:** An exclusive, optional feature. System Secure Sockets Layer (SSL) Security Level 3 works in conjunction with the System SSL component of the Cryptographic Services base element to provide stronger encryption (greater than 64 bits) than that available without this feature. This feature uses the RC2/RC4, TDES, and Advanced Encryption Standard (AES) algorithms for encryption.
- ▶ **Text Search:** An exclusive, base element. Text Search is a database and Web search engine. It has two components:
  - IBM Text Search Engine is a database search engine that is also used in several other IBM products, such as Intelligent Miner™ for Text and DB2 Text Extenders.
  - NetQuestion Solution extends the IBM Text Search Engine into a search engine for z/OS and z/OS.e Web servers.
- ▶ **TIOC:** An exclusive, base element that allows console services and TSO/E to communicate with the terminal hardware.
- ▶ **3270 PC File Transfer Program:** A base element that transfers files from the host to the workstation for offline data manipulation, updating, or correction or for the transfer and storage of local data in the host system.

## 1.10 z/OS Security Server: RACF component

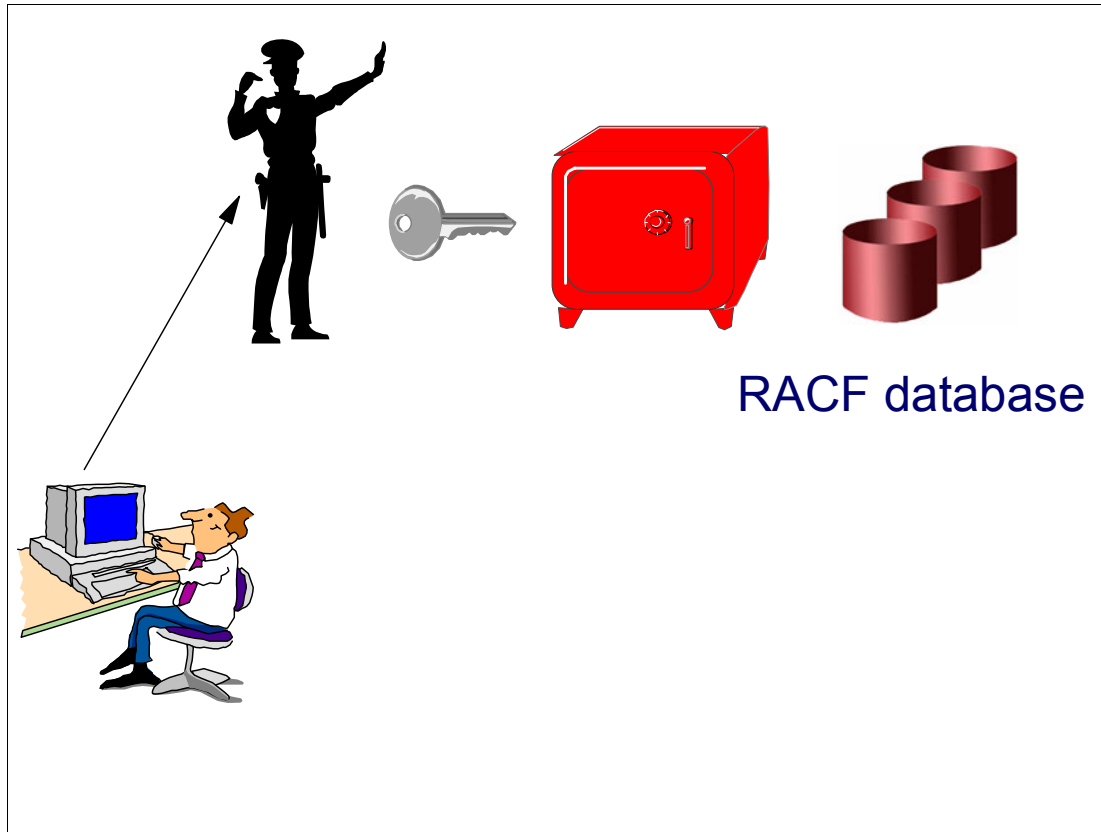


Figure 1-10 z/OS Security Server (RACF component)

The z/OS Security Server is the IBM security product. The RACF product is a component of the z/OS Security Server and works together with the existing system features of z/OS to provide improved data security for an installation. If this product is to be installed in your environment, then RACF customization must be done.

RACF helps meet the need for security by providing:

- ▶ Flexible control of access to protected resources
- ▶ Protection of installation-defined resources
- ▶ Ability to store information for other products
- ▶ Choice of centralized or decentralized control of profiles
- ▶ An ISPF panel interface
- ▶ Transparency to end users
- ▶ Exits for installation-written routines

In order for RACF to meet the specific requirements of your installation, you can customize functions to take advantage of new support after the product is installed. For example, you can tailor RACF through the use of installation exit routines, class descriptor table (CDT) support, or options to improve performance.

## 1.11 Data Facility Storage Management Subsystem (DFSMS)

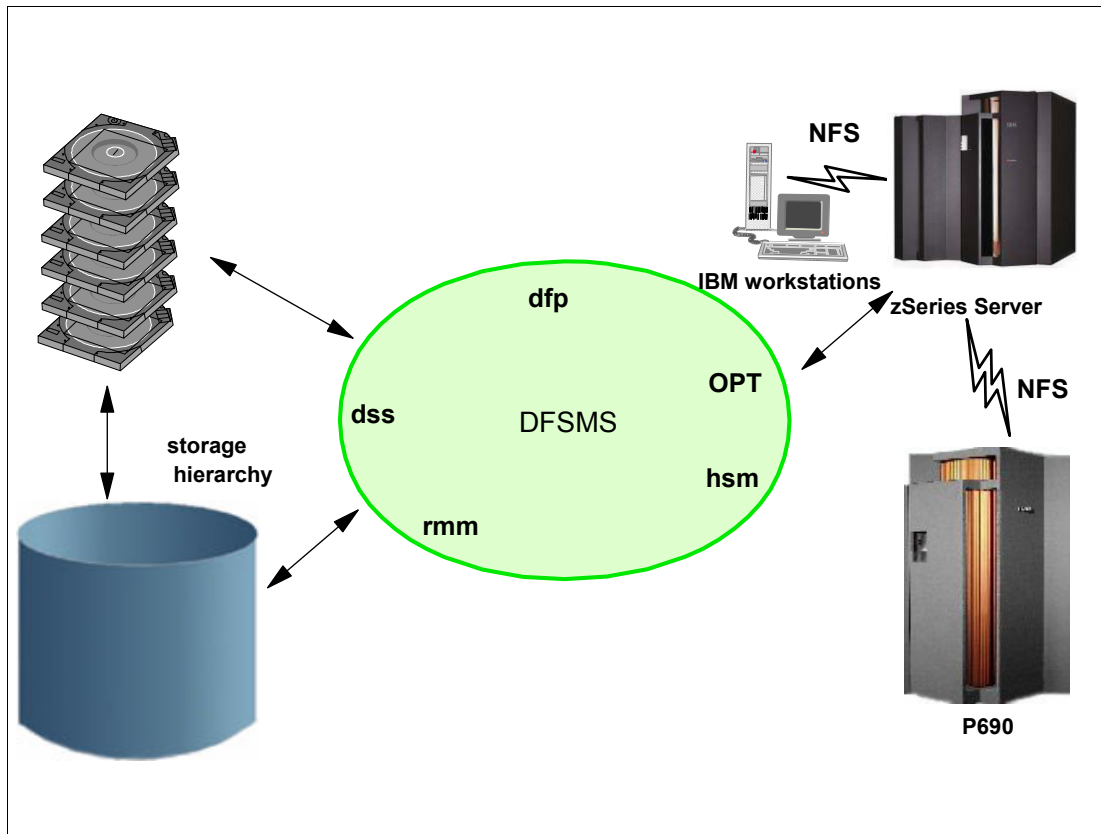


Figure 1-11 Data Facility Storage Management Subsystem

DFSMS provides a range of automated data and space management functions that eliminate, simplify, and automate tasks normally done by users or a storage administrator; improve storage space use; control external storage centrally; and let the storage administrator manage storage storage growth.

DFSMS makes it easier to convert to new device types and takes advantage of what available hardware can do. DFSMS is a family of products, each one having specific functions. The products and functions are as follows:

1. DFSMSdfp™ is part of the z/OS base elements. Together with BCP it forms the foundation of the z/OS operating system, performing the essential data, storage, and device management functions of the system. Data Facility Product (dfp) is in charge of:
  - Space allocation
  - Access methods
  - Support for the storage hardware to balance performance throughout the system
  - Program management tools
  - Applying the storage management policy throughout the SMS constructs
2. DFSMSdss, the Data Set Services (dss), is a high speed and high capacity utility used to move data from disk to disk (copy) or disk to tape (dump) very quickly and efficiently. It performs both logical dumps (to copy data) and physical dumps (to copy track images). DFSMSdss is a external interface to Concurrent Copy, SNAP/SHOT® functions,

explained later. DFSMSdss is also used by other DFSMS components to perform their functions.

3. Hierarchical Storage Manager, DFSMSHsm, provides the following functions:

- Full volume dump and restore
- Policy-based space management
- Automatic and periodic data backup data set and volume levels
- Data set backup (full and incremental)
- Data set recovery
- Aggregate backup and recovery support (ABARS)
- Automatic or user-initiated migration and recall data sets from HSM database (disk or tape)

ABARS allows installations to define an aggregation of data to be backed up and restored as a logical entity. ABARS maintains the point in time relationship of the data within a specific aggregate group. In case of a disaster, when the application is brought online at the recovery location, the data is synchronized and the application can be started. The scope of the aggregation of data that is defined to ABARS is typically that of a critical application or applications. ABARS finds where the data exists within the DFSMS storage device hierarchy (including user disk, user tape, DFSMSHsm migration disk or tape volumes, and so forth) and packages those data into one to four output files that can be physically or electronically sent to an off-site location. ABARS also collects the meta data associated with the backed up data, such as catalog information, Generation Data Group base information, or DFSMSHsm control data set records for migrated data sets, and restores them along with the data.

4. DFSMSrmm™ (Removable Media Manager) is a full-function construct-driven tape management feature that provides the following functions:

- Manages tape volumes and data sets in systems (both automated and manual)
- Keeps track of the locations where tapes are kept
- Policy-driven Library Management

DFSMSrmm can manage:

- A removable media library, which incorporates all other libraries, such as:
  - System-managed tape libraries; for example, the automated IBM TotalStorage® Enterprise Automated Tape Library (3494) and IBM TotalStorage Virtual Tape Servers (VTS)
  - Non-system-managed tape libraries or traditional tape libraries
- Storage locations that are on-site and off-site
- Storage locations defined as home locations

5. DFSMStvs (Transactional VSAM Services) enables batch jobs and CICS online transactions to update shared VSAM data sets concurrently.

Except for DFSMSdfp, which is a base element, all others features are exclusive and optional.

## 1.12 DFSMS Extended Remote Copy: XRC

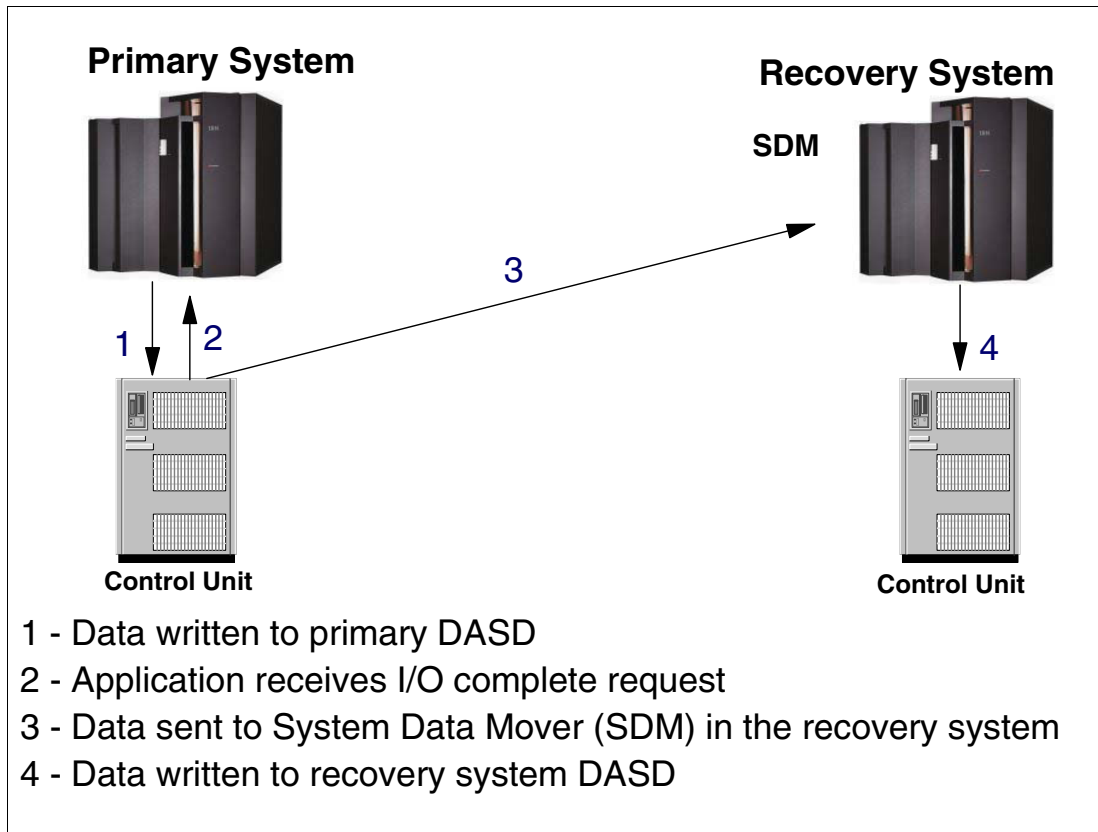


Figure 1-12 DFSMS Extended Remote Copy

DFSMS provides functions that aid installations in making copies of production data for the purpose of business continuance. DFSMS contains a software function called System Data Mover (SDM) that, when combined with the appropriate microcode on a disk storage subsystem (IBM or not), provides an extended distance remote copy capability called XRC.

XRC offers the highest levels of continuous data availability in a disaster recovery and workload movement environment. XRC provides asynchronous remote copy (mirror) of critical data over long distances to a second remote location with minimal impact.

Peer-to-Peer Remote Copy (PPRC) is a hardware solution for rapid and accurate disaster recovery as well as a solution to workload and DASD migration. Updates made on the primary DASD volumes are synchronously shadowed to the secondary DASD volumes.

Besides the remote copy services functions, there are three disk copy services that DFSMS provides:

- ▶ **Concurrent Copy (CC):** Concurrent Copy is a storage subsystem extended function that can generate a copy or a dump of data while applications are updating those data. CC uses DFSMSdss and works with SDM to control the process. DFSMShsm uses the CC feature when invoking DFSMSdss during its backup or dump processing.
- ▶ **FlashCopy®:** FlashCopy is a point-in-time copy services function that can quickly copy data from a source location to a target location.
- ▶ **SnapShot:** SnapShot is a point-in-time copy services function that allows you to snap (quickly copy) data directly from the source location to a target location.

## 1.13 z/OS Communications Server: TCP/IP

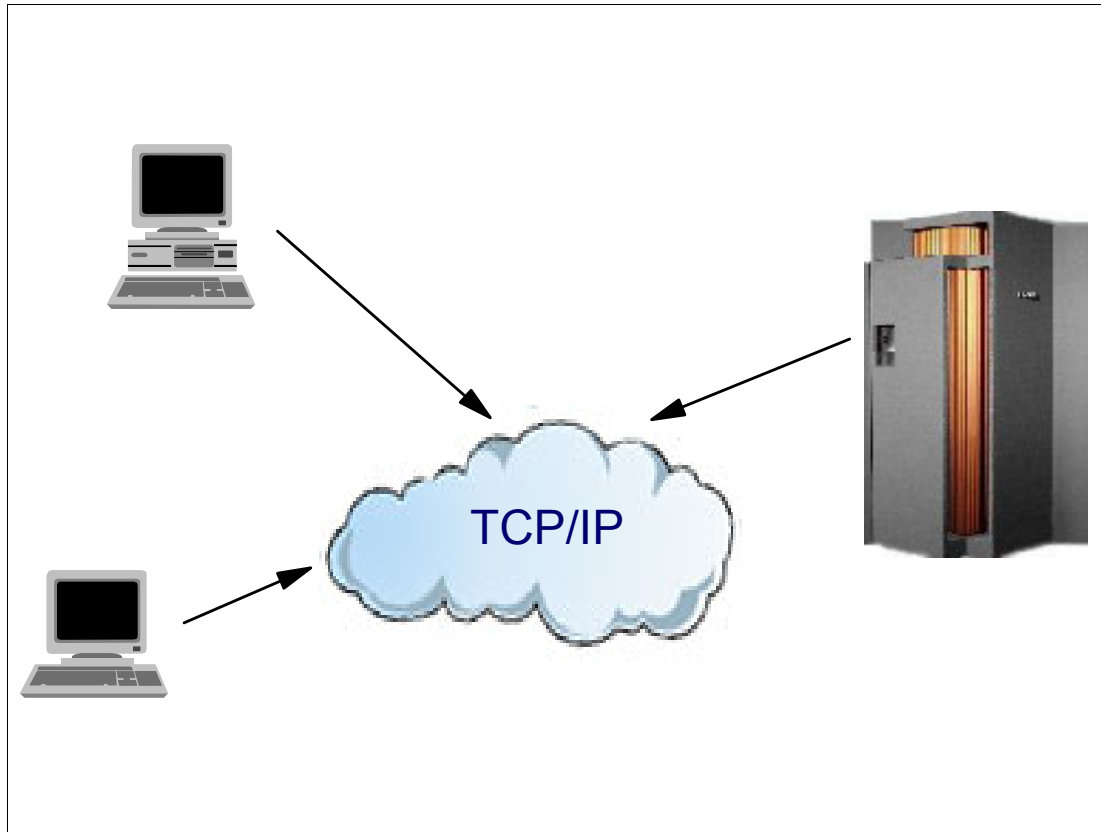


Figure 1-13 Transmission Control Protocol/Internet Protocol (TCP/IP)

Transmission Control Protocol/Internet Protocol (TCP/IP) is a set of protocols and applications that allow you to perform certain computer functions in a similar manner independent of the types of computers or networks being used. When you use TCP/IP, you are using a network of computers to communicate with other users, share data with each other, and share the processing resources of the computers connected to the TCP/IP network.

A computer network is a group of computer nodes electronically connected by some communication medium. Each node has the hardware and the programs necessary to communicate with other computer nodes across this communication medium. The node can be a PC, workstation, microcomputer, departmental computer, or large computer system. The size of the computer is not important, while the ability to communicate with other nodes is.

Computer networks allow you to share the data and computing resources of many computers. Applications, such as departmental file servers, rely on networking as a way to share data and programs.

Many forms of communication media are available today. Each is designed to take advantage of the environment in which it operates. Communication media consist of a combination of the physical network used to connect to computer nodes and the language, or protocol, they use to communicate with each other.



## 1.14 System Modification Program Extended (SMP/E)

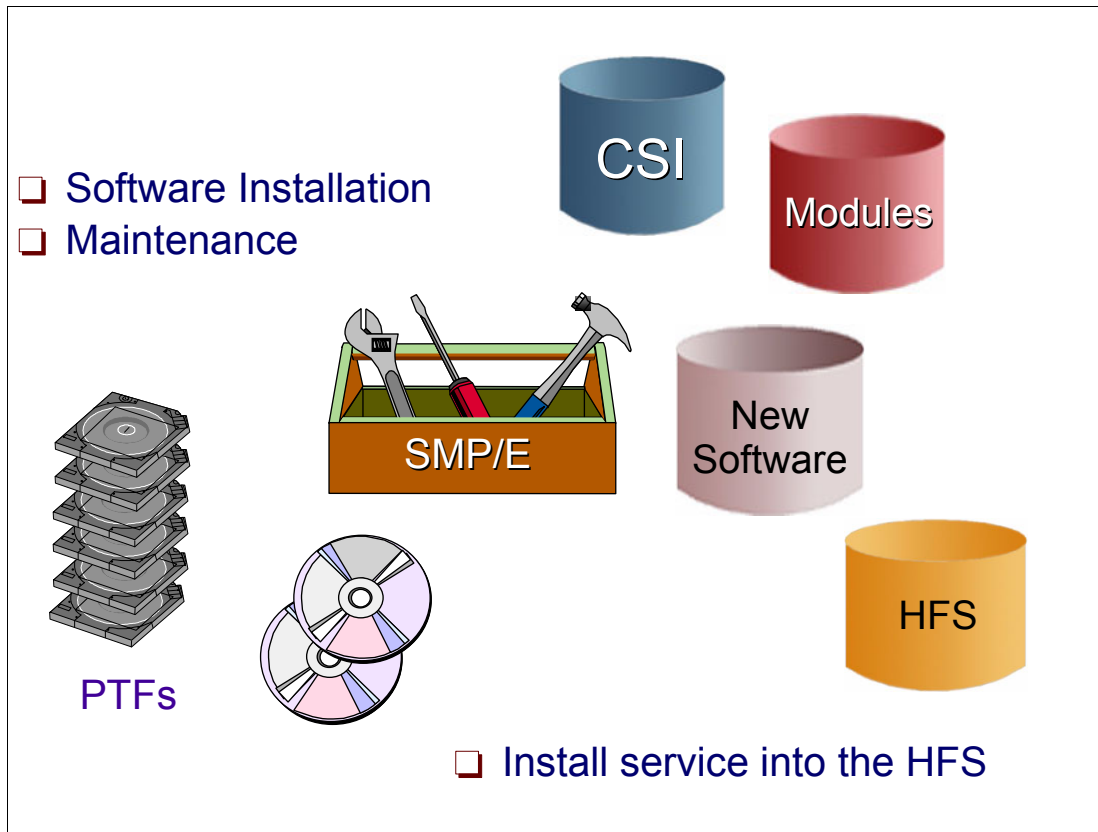


Figure 1-14 System Modification Program Extended

System Modification Program Extended (SMP/E) is a tool designed to manage the installation of software products on your z/OS system, and to track the modifications applied to those products. Usually, it is the system programmer's responsibility to ensure that all software products and modifications are properly installed on the system, and that all products are installed at the proper level so all elements of the system can work together.

A z/OS system may appear to be one big block of code that drives the CPU. Actually, z/OS is a complex system comprising many different smaller blocks of code. Each of those smaller blocks of code performs a specific function within the system. Each system function is composed of one or more load modules. In a z/OS environment, a load module represents the basic unit of machine-readable executable code. Load modules are created by combining one or more object modules and processing them with a link-edit utility. The link-editing of modules is a process that resolves external references and addresses. The functions on a system, therefore, are one or more object modules that have been combined and link-edited.

Over time, you may need to change some of the elements of your system. These changes may be necessary to improve the usability or reliability of a product. You may need to add some new functions to your system, upgrade some of the elements of your system, or modify some elements for a variety of reasons. In all cases, you are making system modifications.

All executable software code is subject to errors. Any errors in the IBM software code are fixed by IBM and made available to installations in the form of either an authorized program analysis report (APAR) or program temporary fix (PTF).

## 1.15 Infoprint Server

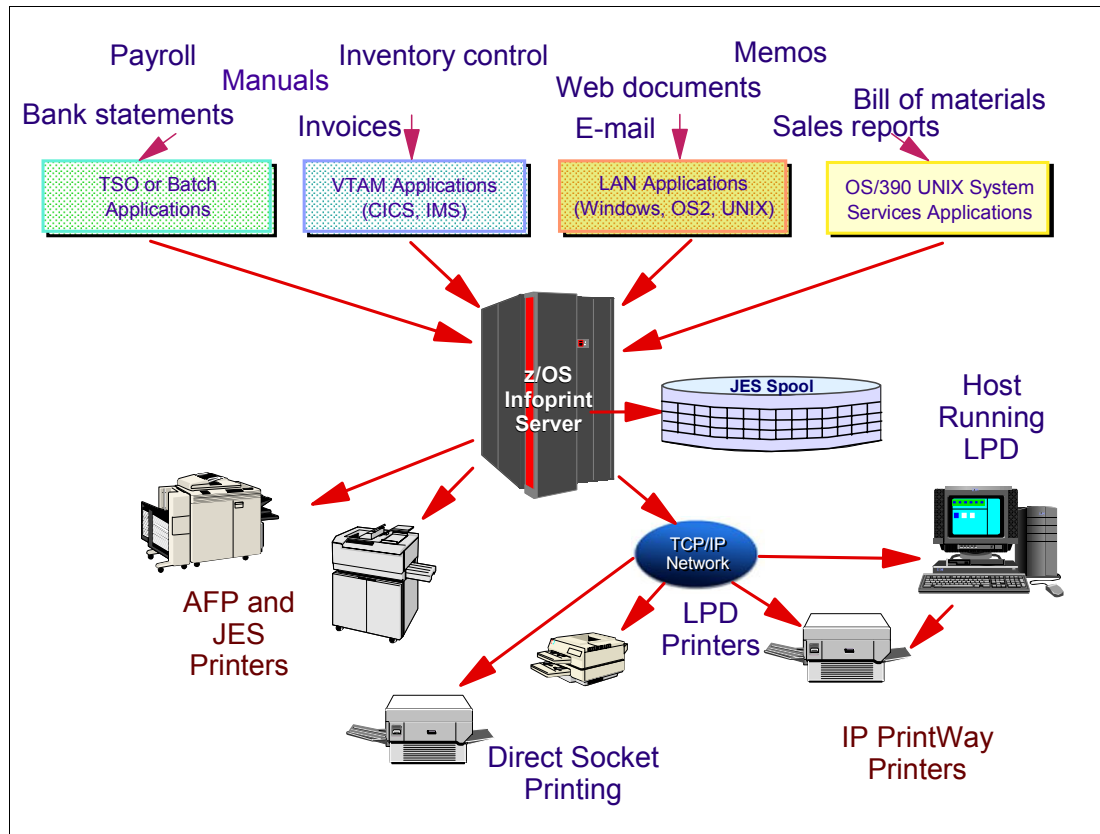


Figure 1-15 Infoprint functions and features

Infoprint Server uses the z/OS JES spool, a powerful print spooler, to manage the printers on z/OS. In addition to the JES spool's traditional functions of scheduling and recovering print jobs, Infoprint Server has enhanced the use of the JES spool in these ways:

- ▶ Users can submit more data streams to the JES spool, specifically:
  - PostScript
  - Printer Control Language (PCL)
  - American National Standard Code for Information Interchange (ASCII)
  - Data from VTAM applications (such as CICS and IMS) that traditionally print on SNA printers
- ▶ If you install Infoprint Server Transforms (5697-F51), users can also submit these data streams:
  - Portable Document Format (PDF)
  - SAP Output Text Format (OTF)
  - SAP Advanced Business Application Programming (ABAP)
- ▶ If you install Infoprint XML Extender for z/OS (5655-J66), users can also submit the data stream:
  - Extensible Markup Language (XML)

Files on the JES spool can be printed not only on the traditional array of JES-controlled printers and printers driven by Print Services Facility™ (PSF) for z/OS, but also on ASCII printers in a TCP/IP network and on VTAM-controlled printers in an SNA network. Using Infoprint Server, files on the JES spool can also be sent over the Internet to IPP-enabled printers and to e-mail destinations.

## 1.16 Resource Management Facility (RMF)

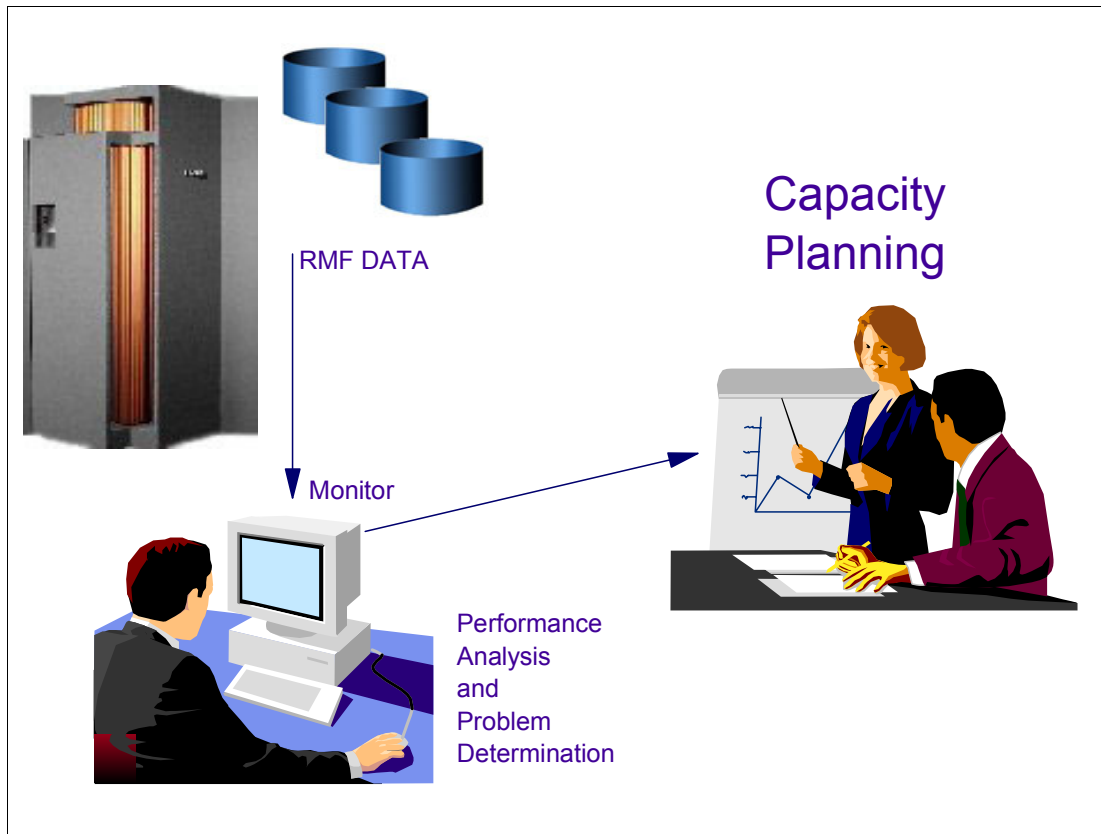


Figure 1-16 Resource Measurement Facility

Many different activities are required to keep your z/OS running smoothly, and to provide the best service on the basis of the available resources and workload requirements. The console operator, the service administrator, the system programmer, or the performance analyst does these tasks. RMF is the tool (online and batch reports) that helps each of these people do the job effectively.

RMF gathers data using three monitors:

- ▶ Short-term data collection with Monitor III
- ▶ Snapshot monitoring with Monitor II
- ▶ Long-term data gathering with Monitor I

Data is gathered for a specific cycle time, and consolidated data records are written at a specific interval time. The default value for data gathering is one second and for data recording 30 minutes. You can select these options according to your requirements and change them whenever the need arises.

Monitor I collects long-term data about system workload and resource utilization, and covers all hardware and software components of your system: processor, I/O device and storage activities and utilization, as well as resource consumption, activity, and performance of groups of address spaces.

## 1.17 System Management Facility (SMF)

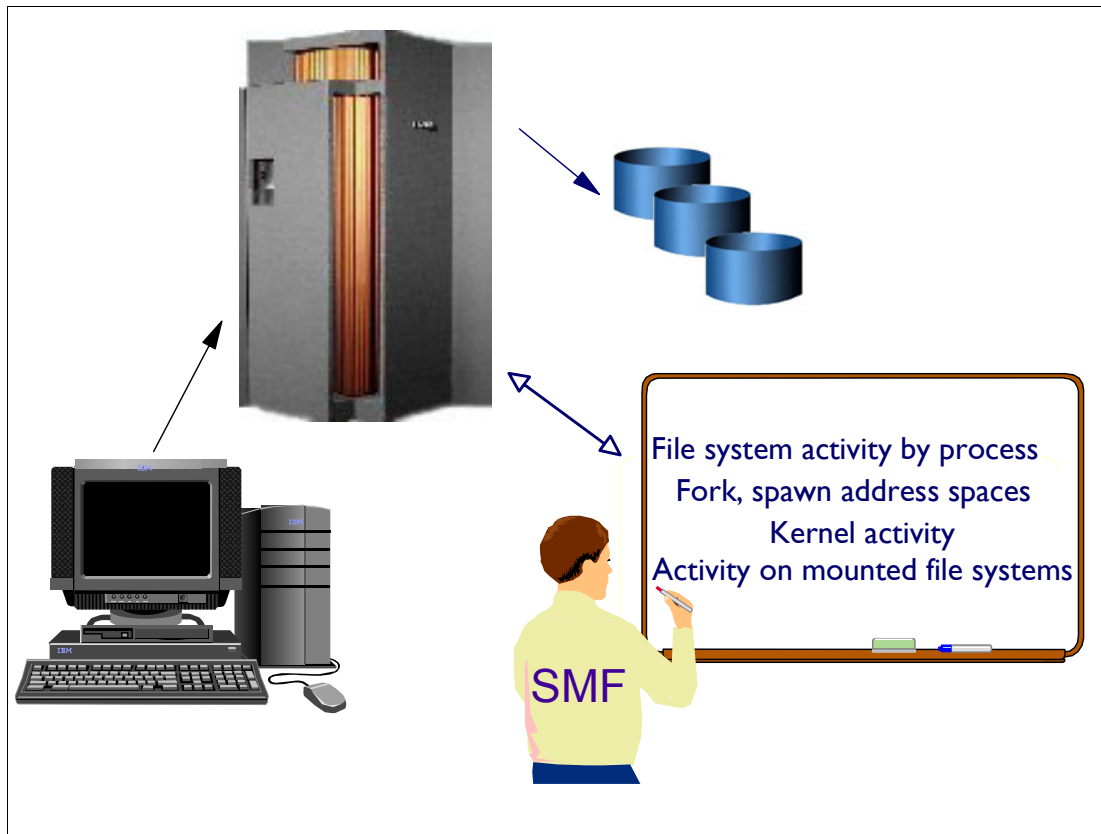


Figure 1-17 System Management Facility

System management facility (SMF) collects and records system and job-related information that your installation can use in:

- ▶ Billing users
- ▶ Reporting reliability
- ▶ Analyzing the configuration
- ▶ Scheduling jobs
- ▶ Summarizing direct access volume activity
- ▶ Evaluating data set activity
- ▶ Profiling system resource use
- ▶ Maintaining and auditing system security

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 on the CPU time, SYSOUT activity, and data set activity of each job step, job, APPC/MVS transaction program, and TSO/E session.

An installation can provide its own routines as part of SMF. These routines will receive control either at a particular point as a job moves through the system, or when a specific event occurs. For example, an installation-written routine can receive control when the CPU time limit for a job expires or when an initiator selects the job for processing. The routine can collect additional information, or enforce installation standards.

## 1.18 Time Sharing Option/Extended (TSO/E)

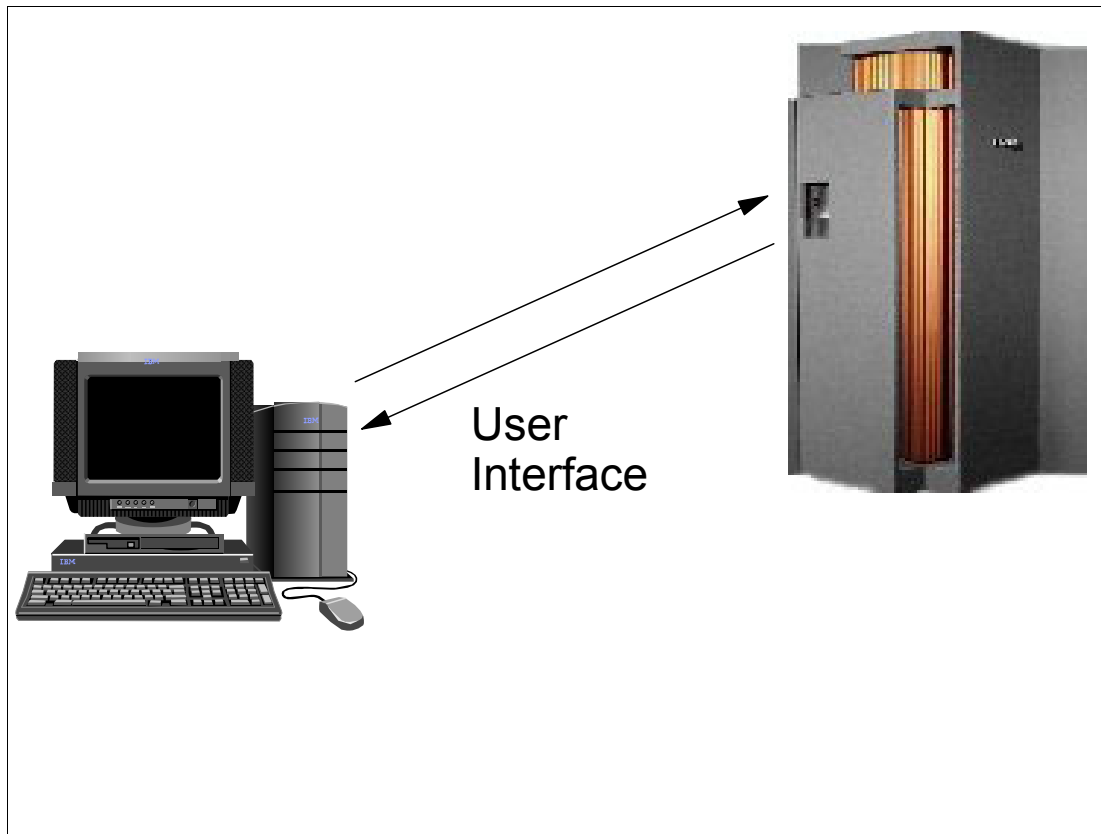


Figure 1-18 Time Sharing Option/Extended

TSO/E is a base element of the z/OS operating system that allows users to interactively work with the system. In general, TSO/E makes it easier for people with all levels of experience to interact with the z/OS system.

TSO/E has advantages for a wide range of computer users. TSO/E users include system programmers, application programmers, information center administrators, information center users, TSO/E administrators, and others who access applications that run under TSO/E.

You can use TSO/E in any one of the following three environments:

- ▶ **Line Mode TSO/E:** Using TSO/E commands typed on a terminal, one line at a time. This is a quick and direct way to use TSO/E and was the way programmers originally used to communicate interactively with the z/OS operating system.
- ▶ **ISPF/PDF:** The Interactive System Productivity Facility (ISPF) and its Program Development Facility (ISPF/PDF) work together with TSO/E to provide panels with which users can interact. ISPF provides dialog management service that displays panels and enables a user to navigate through the panels. ISPF/PDF is a dialog of ISPF that helps maintain libraries of information in TSO/E and allows a user to manage the library through facilities such as browse, edit, and utilities.
- ▶ **Information Center facility:** A set of panels that enable you to display services like mail and names directory, perform data analysis, and prepare documents such as reports, graphs, and charts.

## 1.19 UNIX System Services

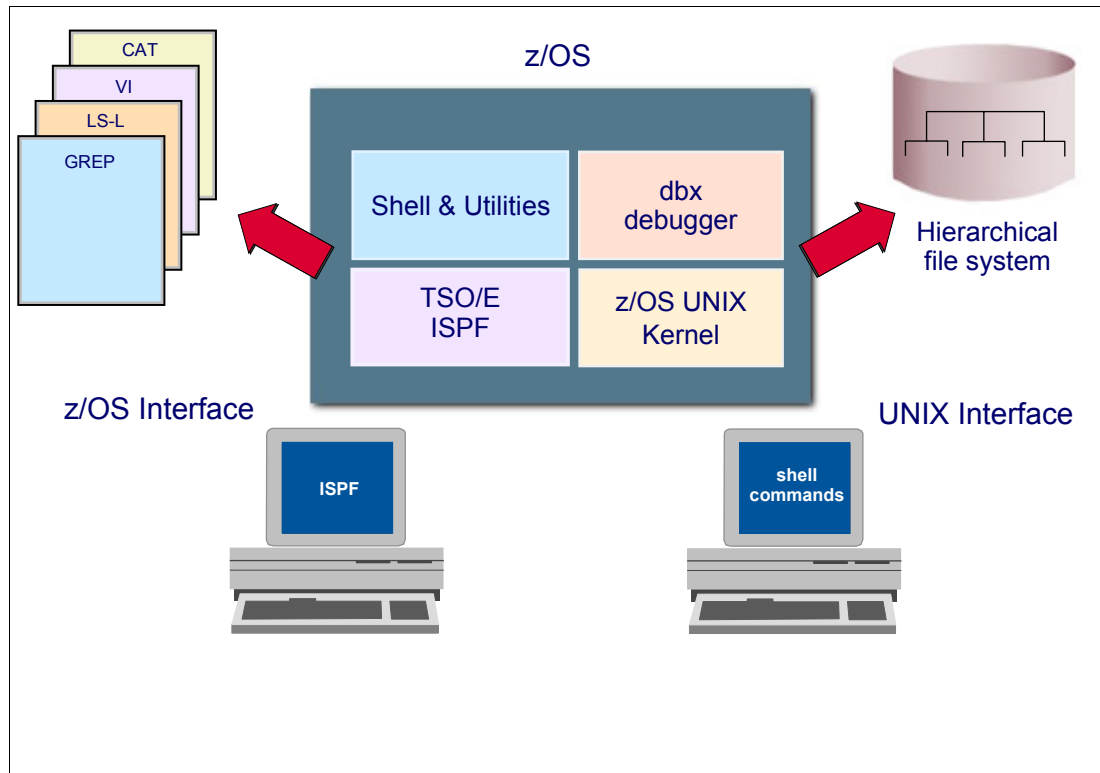


Figure 1-19 UNIX System Services

Beginning with OS/390 V2R3, UNIX System Services has been merged with the BCP, and is now part of the BCP FMID. In addition, the OMVS address space is started automatically. BPXOINIT is the started procedure that runs the initialization process. The OMVS address space is now started automatically at IPL by means of the OMVS statement in the IEASYSxx SYS1.PARMLIB member.

z/OS UNIX interacts with the following elements and features of z/OS:

- ▶ C/C++ Compiler, to compile programs
- ▶ Language Environment, to execute the shell and utilities
- ▶ Data Facility Storage Management Subsystem
- ▶ z/OS Security Server
- ▶ Resource Measurement Facility (RMF)
- ▶ System Display and Search Facility (SDSF)
- ▶ Time Sharing Option Extensions (TSO/E)
- ▶ TCP/IP Services
- ▶ ISPF, to use the dialogs for OEDIT, or ISPF/PDF for the ISPF shell
- ▶ BookManager READ, to use the OHELP online help facility

z/OS UNIX System Services provides APIs compatible with industry standards X/Open Portability Guide, Issue 4 (XPG4\*\*).

## 1.20 Some z/OS services requiring customization



*Figure 1-20 Some z/OS services requiring system programmer customization*

Some z/OS services require customization; they are listed in Figure 1-20. In the following sections we discuss these services in detail.

## 1.21 Library Lookaside (LLA)

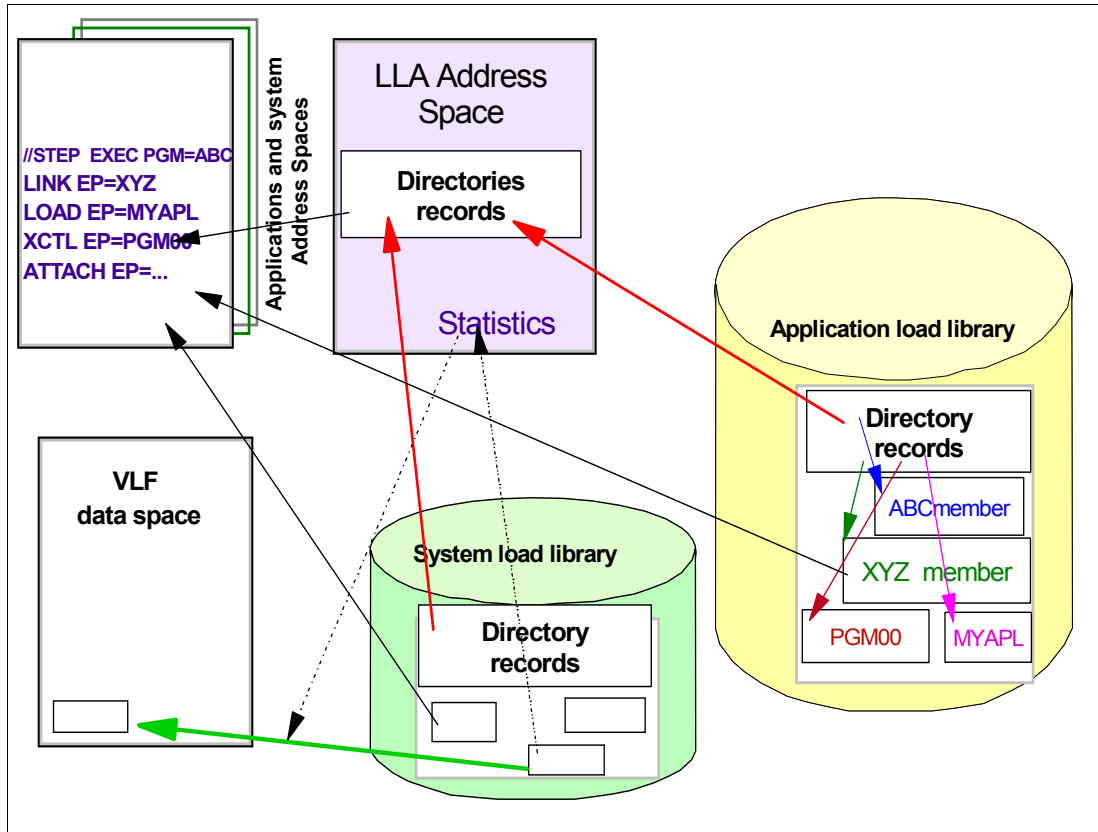


Figure 1-21 Library Lookaside

Library Lookaside (LLA) provides services that improve the system performance by reducing the amount of I/O needed to locate and fetch modules from DASD storage. LLA services are in LLA address space, a started task, and improve module fetch performance as follows:

- ▶ LLA maintains, in LLA address space, copies of the library directories. To fetch a module, the system has to first search the directory for the module location. The system can quickly search the LLA copy of a directory in virtual storage instead of using costly I/O to search the directories on DASD.
- ▶ LLA places copies of selected modules in a Virtual Lookaside Facility (VLF) data space (when the LLA class is defined to VLF). The system can quickly fetch modules from virtual storage, rather than using slower I/O to fetch the modules from DASD.
- ▶ LLA determines which modules, if staged, would provide the most benefit to module fetch performance. LLA evaluates modules as candidates for staging based on statistics LLA collects about the members of the libraries it manages, such as module size, fetch count, and the time required to fetch a particular module.

The benefits of LLA apply only to modules that are retrieved through the macros LINK, LINKX, LOAD, ATTACH, ATTACHX, XCTL, and XCTLX. Directory entries for the primary system library, program libraries concatenated to it in SYS1.PARMLIB(LNKLSTxx), and additional production libraries named in SYS1.PARMLIB(CSVLLAxx) are read into the private area of the LLA address space during its initialization. Subsequent searches for programs in these libraries begins with the directories in LLA, and not in the directories on DASD.

You obtain the most benefit from LLA when you have both LLA and VLF functioning, so you should plan to use both.



## 1.22 Virtual Lookaside Facility (VLF)

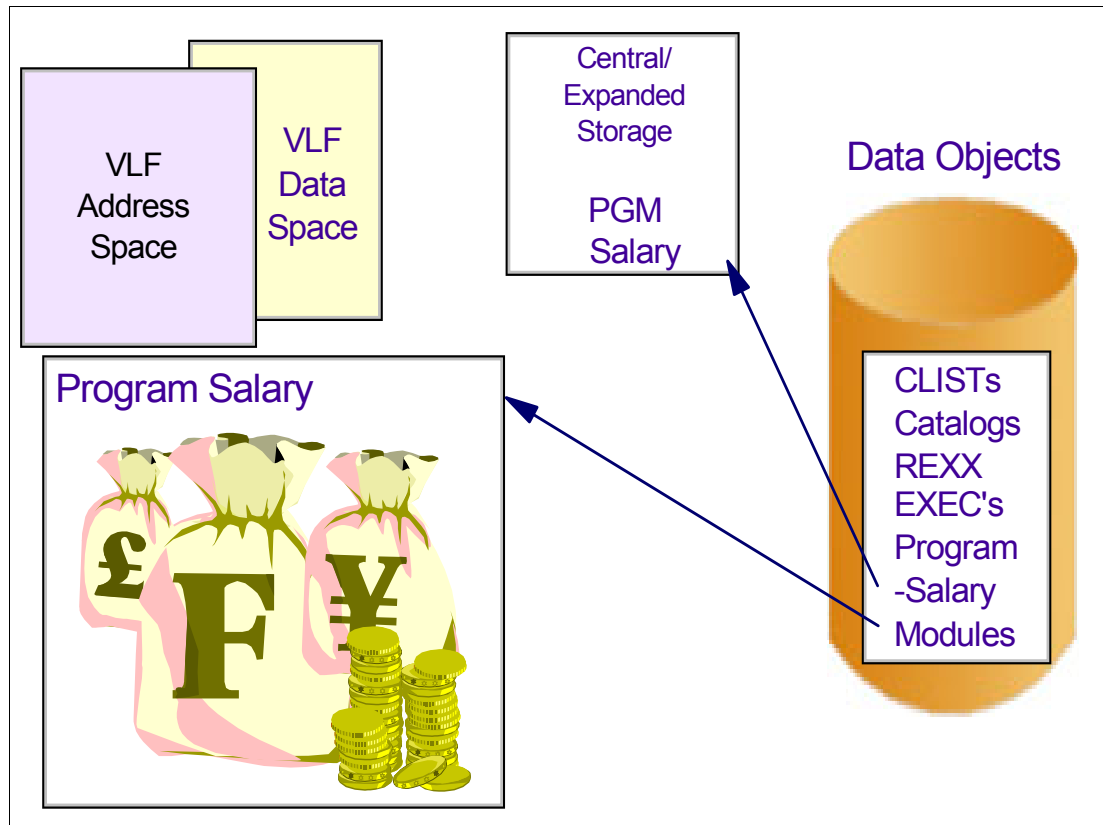


Figure 1-22 Virtual Lookaside Facility (VLF)

Virtual Lookaside Facility (VLF) is a set of services that can improve the response time of applications that must retrieve a set of data for many users. VLF creates and manages a data space to store an application's most frequently used data. When the application makes a request for data, VLF checks its data space to see if the data is there. If the data is present, VLF can rapidly retrieve it without requesting I/O to DASD.

To take advantage of VLF, an application must identify the data it needs. The data is known as a data object. Data objects should be small to moderate in size, named according to the VLF naming convention, and associated with an installation-defined class of data objects.

Certain IBM products or components such as LLA, TSO/E, CAS, and RACF use VLF as an alternate way to access data. Since VLF uses virtual storage for its data spaces, there are performance considerations each installation must weigh when planning for the resources required by VLF.

**Note:** VLF is intended for use with major applications. Because VLF runs as a started task that the operator can stop or cancel, it cannot take the place of any existing means of accessing data on DASD. Any application that uses VLF must also be able to run without it.

When you define the LLA class to VLF, and start VLF, the most active modules from LLA-managed libraries are staged into the DCSVLLA data space managed by VLF. You obtain the most benefit from LLA when you have both LLA and VLF functioning, so you should plan to use both.

## 1.23 Workload Manager (WLM)

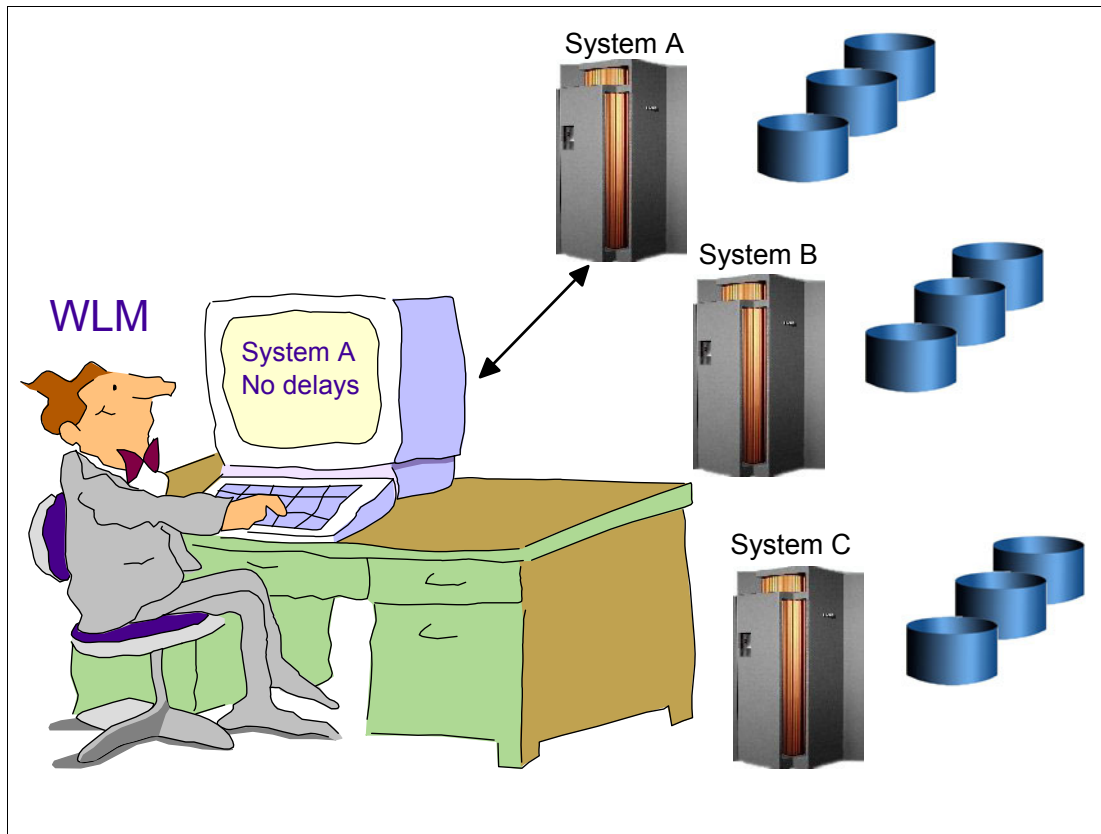
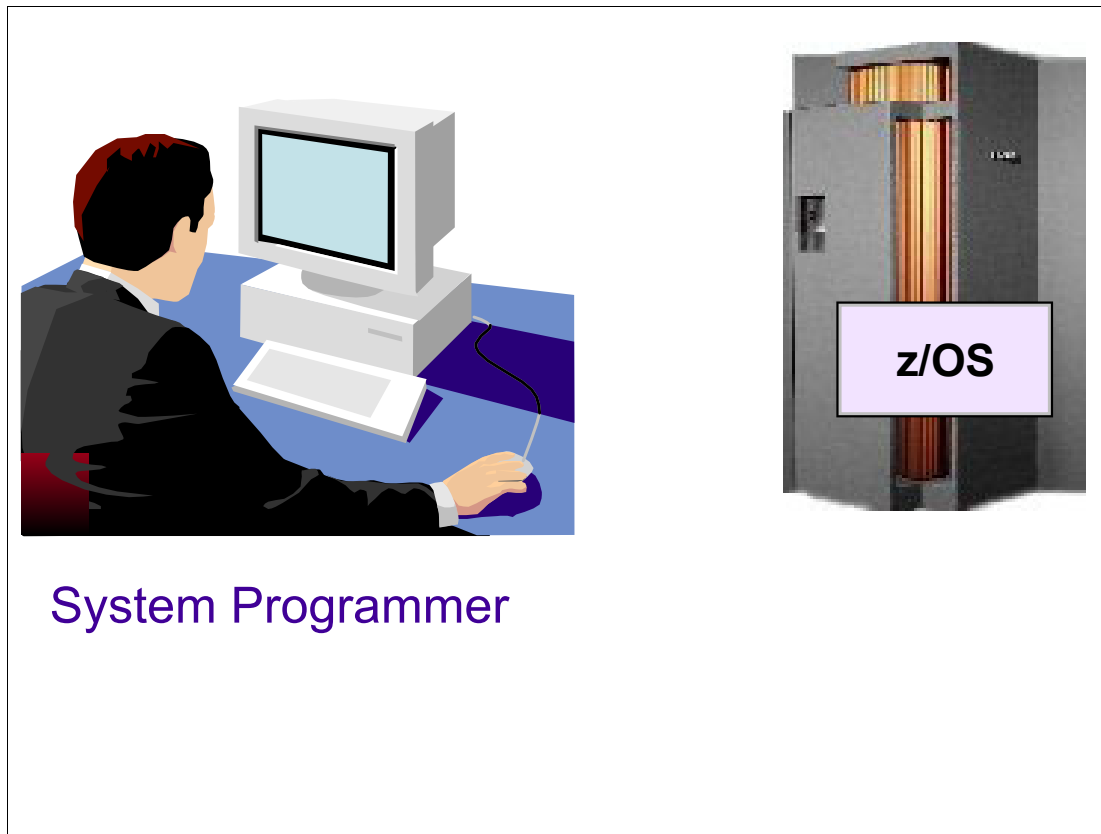


Figure 1-23 Workload Manager

Before the introduction of Workload Manager (WLM), the only way to inform the z/OS about your business goal, such as response time, was to translate from high-level objectives about what work needs to be done into the extremely technical terms that the system can understand. This translation required highly skilled staff, and could be protracted, error-prone, and eventually in conflict with the original business goals. Additionally, it was often difficult to predict the effects of changing a parameter, which may be required, for example, following a system capacity increase. This could result in unbalanced resource allocation, that is, feeding work one resource while starving it of another. This way of operation, known as *compatibility mode*, was becoming unmanageable as new workloads were introduced, and as multiple systems were being managed together in parallel sysplex processing and data sharing environments.

When in *goal mode* system operation, WLM provides fewer, simpler, and more consistent system externals that reflect goals for work expressed in terms commonly used in business objectives, and WLM and Service Request Manager (SRM) match resources to meet those goals by constantly monitoring and adapting the system. Workload Manager provides a solution for managing workload distribution, workload balancing, and distributing resources to competing workloads.

## 1.24 z/OS operating system: the role of a system programmer



*Figure 1-24 z/OS Operating System: the role of a system programmer*

To meet the specific requirements of your installation, you can customize z/OS functions and interfaces to take advantage of new functions after installation. The role of the system programmer is to install, customize, and maintain the operating system. The z/OS operating system runs on various hardware configurations. A system programmer must define the hardware I/O configuration resources that are to be available to the z/OS operating system. The hardware used can be either IBM or other manufacturer machines. As a z/OS system programmer, you must be aware of the following:

- ▶ Storage concepts
- ▶ Virtual storage and address spaces concepts
- ▶ Device I/O configurations
- ▶ Processor configurations
- ▶ Console definitions
- ▶ System libraries where the software is placed
- ▶ System data sets and their placement
- ▶ Customization parameters that are used to define your z/OS configuration

## 1.25 z/OS system programmer management overview

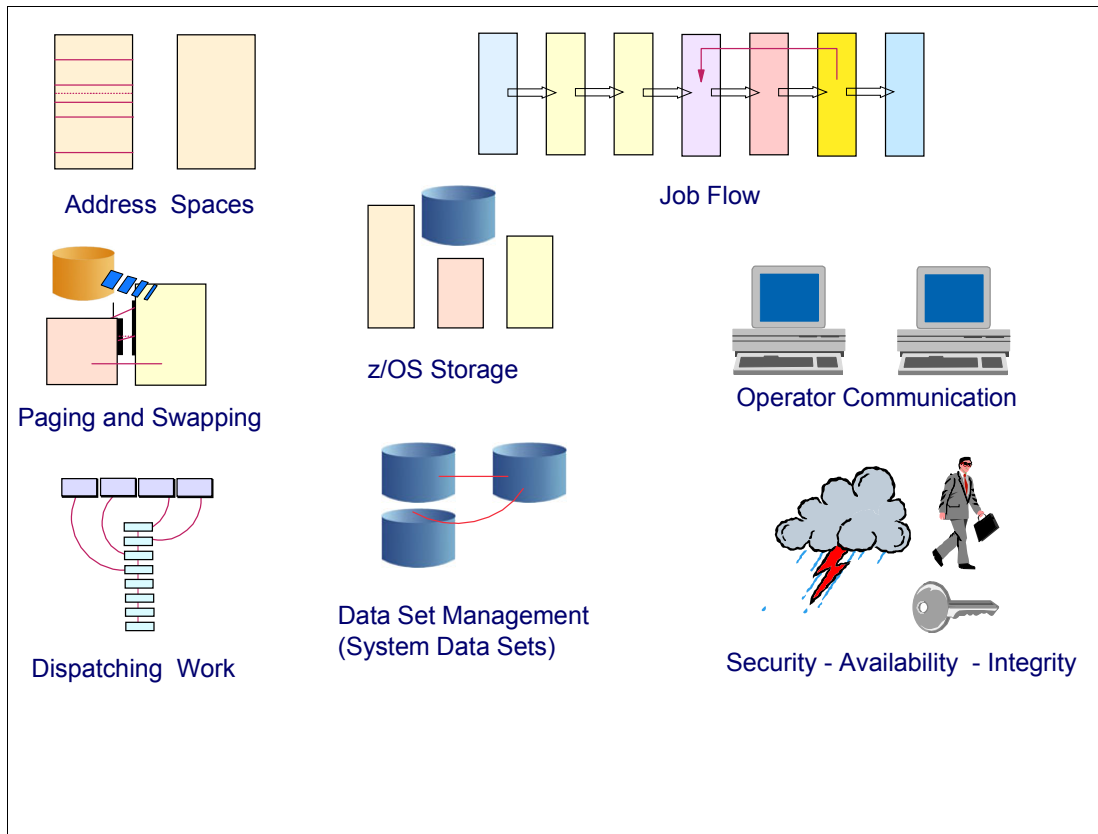


Figure 1-25 z/OS system programmer management overview

As a z/OS system programmer, you need to be involved in the customization of the items shown on Figure 1-25. These items are as follows:

- ▶ **Address spaces:** When you start the z/OS operating system, z/OS establishes system component address spaces. During the IPL, the first address space started is the master scheduler address space (\*MASTER\*). There are other system address spaces for various subsystems and system components.
- ▶ **Paging:** Page data sets contain the paged-out portions of address spaces, the common service area (CSA), and the data written to virtual I/O (VIO) data sets.
- ▶ **Dispatching work:** The scheduling of address spaces and other tasks to execute in the z/OS system is done by the z/OS dispatcher. The z/OS dispatcher performs two major system functions. It is responsible for finding and dispatching the highest priority unit of work in the system (SRB, Task, or Interrupted Local Supervisor Routine), and saving status for locked and unlocked tasks and SRBs.
- ▶ **Job flow:** z/OS uses a job entry subsystem (JES) to receive jobs into the operating system, to schedule them for processing by z/OS, and to control their output processing. JES is the component of the operating system that provides supplementary job management, data management, and task management functions such as scheduling, control of job flow, and spooling.
- ▶ **z/OS storage:** The system programmer must be aware of all storage considerations when installing and customizing a z/OS operating system environment. The initialization process begins when the system operator selects the LOAD function at the system console. z/OS locates all of the usable central storage that is online and available to the

system, and creates a virtual environment for the building of various system areas. This initialization phase allocates the system's minimum virtual storage for the system queue area (SQA) and the extended SQA, allocates virtual storage for the extended local system queue area (extended LSQA) for the master scheduler address space, and allocates virtual storage for the common service area (CSA) and the extended CSA. The amount of storage allocated depends on the values specified on the CSA system parameter at IPL.

- ▶ **System data sets:** Each installation must incorporate required system data sets into the system by allocating space for them on appropriate direct access devices during system installation. The DEFINE function of Access Method Services is used to define both the space requirements and the volume for each system data set. Some data sets must be allocated on the system residence volume, while some can be placed on other direct access volumes.
- ▶ **Operator communication:** The operation of a z/OS system involves the following:
  - Console operations, or how operators and system programmers interact with z/OS to monitor or control the hardware and software
  - Message and command processing that forms the basis of operator interaction with z/OS and the basis of z/OS automation
  - Managing hardware such as processors and peripheral devices (including the consoles where operators or system programmers do their work) and software such as the z/OS operating control system, the job entry subsystem, subsystems such as NetView® that can control automated operations, and all the applications that run on z/OS
- ▶ **Security:** Data security is the protection of data against unauthorized disclosure, transfer, modification, or destruction, whether accidental or intentional. A security system must be installed in your operating system by a system programmer to maintain the resources necessary to meet the security objectives. The system programmer has the overall responsibility, using the technology available, to transform the objectives of the security policy into a usable plan.
- ▶ **Availability:** The software products supporting system programmers and operators in managing their systems heavily influence the complexity of their job and their ability to keep system availability at a high level. Performance management is the system management discipline that most directly impacts all users of system resources in an enterprise. You can do this with RMF.
- ▶ **Integrity:** An operating system is said to have system integrity when it is designed, implemented and maintained to protect itself against unauthorized access, and does so to the extent that security controls specified for that system cannot be compromised. Specifically for z/OS, this means that there must be no way for any unauthorized program, using any system interface, defined or undefined:
  - To bypass store or fetch protection
  - To bypass OS password, VSAM password, or RACF security checking
  - To obtain control in an authorized state

## 1.26 System programmer and z/OS operations

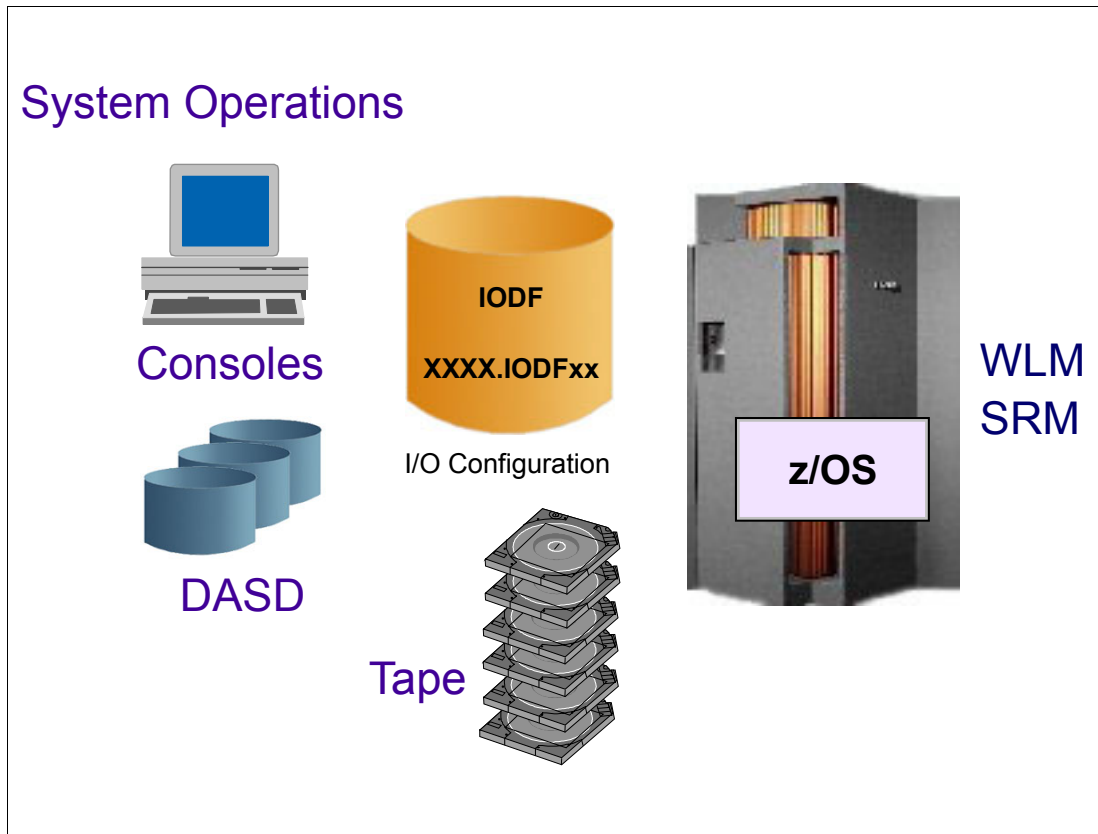


Figure 1-26 System programmer and z/OS operations

A system programmer has to plan the following operations areas:

► **Workload Manager**

Workload Manager provides a solution for managing workload distribution, workload balancing, and distributing resources to competing workloads. Workload Manager is the combined cooperation of various subsystems (CICS, IMS/ESA®, JES, APPC, TSO/E, z/OS UNIX System Services, DDF, DB2, SOM®, LSFM, and Internet Connection Server) with the Workload Manager (WLM) component.

► **System performance**

The task of tuning a system is an iterative and continuous process. The controls offered by SRM are only one aspect of this process. Initial tuning consists of selecting appropriate parameters for various system components and subsystems. Once the system is operational and criteria have been established for the selection of jobs for execution via job classes and priorities, SRM will control the distribution of available resources according to the parameters specified by the installation.

SRM, however, can only deal with available resources. If these are inadequate to meet the needs of the installation, even optimal distribution may not be the answer; other areas of the system should be examined to determine the possibility of increasing available resources.

When requirements for the system increase and it becomes necessary to shift priorities or acquire additional resources, such as a larger processor, more storage, or more terminals, the SRM parameters might have to be adjusted to reflect changed conditions.

► **I/O device management**

You must define an I/O configuration to the operating system (software) and the channel subsystem (hardware). The Hardware Configuration Definition (HCD) component of z/OS consolidates the hardware and software I/O configuration processes under a single interactive end-user interface. The validation checking that HCD does as you enter data helps to eliminate errors before you attempt to use the I/O configuration. The output of HCD is an I/O definition file (IODF), which contains I/O configuration data. An IODF is used to define multiple hardware and software configurations to the z/OS operating system. When you activate an IODF, HCD defines the I/O configuration to the channel subsystem and/or the operating system. With the HCD activate function or the z/OS ACTIVATE operator command, you can make changes to the current configuration without having to initial program load (IPL) the software or power-on reset (POR) the hardware. Making changes while the system is running is known as dynamic configuration or dynamic reconfiguration.

► **Console operations**

The operation of a z/OS system involves the following:

- Console operations or how operators interact with z/OS to monitor or control the hardware and software.
- Message and command processing that forms the basis of operator interaction with z/OS and the basis of z/OS automation.

Operating z/OS involves managing hardware such as processors and peripheral devices (including the consoles where your operators do their work) and software such as the z/OS operating control system, the job entry subsystem, subsystems such as NetView that can control automated operations, and all the applications that run on z/OS.

Planning z/OS operations for a system must take into account how operators use consoles to do their work and how you want to manage messages and commands. Because messages are also the basis of automated operations, understanding message processing in an z/OS system can help you plan z/OS automation.

## **Managing operations**

Also involved are the business goals and policies established to allow the installation to grow and handle work efficiently. These needs, of course, vary from installation to installation, but they are important when you plan your z/OS operations.

Managing the complexity of z/OS requires you to think about the particular needs of the installation. However, any installation might consider the following goals when planning its z/OS operations:

► **Increasing system availability**

Many installations need to ensure that their system and its services are available and operating to meet service level agreements. Installations with 24-hour, 7-day operations need to plan for minimal disruption of their operation activities. In terms of z/OS operations, how the installation establishes console recovery or whether an operator must re-IPL a system to change processing options are important planning considerations.

► **Controlling operating activities and functions**

As more installations make use of multisystem environments, the need to coordinate the operating activities of those systems becomes crucial. Even for single z/OS systems, an installation needs to think about controlling communication between functional areas (such as a tape-pool library and the master console area, for example). In both single and multisystem environments, the commands operators can issue from consoles can be a security concern that requires careful coordination. As a planner, you want to make sure

that the right people are doing the right tasks when they interact with z/OS. If your installation uses remote operations to control target systems, you also need to decide about controlling those activities from the host system.

▶ **Simplifying operator tasks**

Because the complexity of operating z/OS has increased, an installation needs to think about the tasks and skills of its operators. How operators respond to messages at their consoles and how you can reduce or simplify their actions are important to operations planning. Also, your installation needs to plan z/OS operator tasks in relation to any automated operations that help simplify those tasks.

▶ **Streamlining message flow and command processing**

In thinking about operator tasks, an installation needs to consider how to manage messages and commands. Operators need to respond to messages. Routing messages to operator consoles, suppressing messages to help your operators manage increased message traffic, or selecting messages for automated operations can all help you manage system activity efficiently.

▶ **Single system image**

Single system image allows the operator, for certain tasks, to interact with several images of a product as though they were one image. For example, the operator can issue a single command to all z/OS systems in the sysplex instead of repeating the command for each system.

▶ **Single point of control**

Single point of control allows the operator to interact with a suite of products from a single workstation. An operator can accomplish a set of tasks from a single workstation, thereby reducing the number of consoles the operator has to manage.



## 1.27 Requirements for installation

- TSO/E and ISPF
  - Batch job JCL
- Storage concepts
- Device I/O configurations
- Processor configurations
- Console definitions
- System libraries management
- DASD space management
- Customization parameters - SYS1.PARMLIB
- Data set placement

Figure 1-27 Requirements for install

To be able to install and customize a z/OS operating system, a system programmer has to know certain basic skills and functions. Using these skills is documented in this book. Figure 1-27 lists the following areas about which a programmer needs to know:

- |                 |  |
|-----------------|--|
| <b>TSO/E</b>    | TSO/E is Time Sharing Option Extensions. It is an option of the z/OS operating system that allows users to interactively share computer time and resources. TSO/E is an integral part of z/OS, and serves as a platform for other elements, such as BookManager READ, HCD, and ISPF/PDF.   |
| <b>ISPF/PDF</b> | The Interactive System Productivity Facility (ISPF) and its Program Development Facility (ISPF/PDF) work under TSO/E to provide panels with which users can interact. ISPF provides the underlying dialog management service that displays panels and enables a user to navigate through the panels. PDF is a dialog of ISPF that helps maintain libraries of information in TSO/E and allows a user to manage the library through facilities such as browse, edit, and utilities. |
| <b>JCL</b>      | During the install phase of z/OS, many batch jobs are required to be submitted. The JCL for these jobs needs to be updated for your environment. Therefore, it is essential that a system programmer be very familiar with JCL and batch job submission from TSO/E and using ISPF.   |
| <b>Storage</b>  | Storage concepts must be understood by the system programmer in setting up a z/OS environment.   |

- Device I/O** An I/O configuration is the hardware resources available to the operating system and the connections between these resources. The resources include:
- Channels
  - ESCON Directors (switches)
  - Control units
  - Devices
- When you define a configuration, you need to provide both physical and logical information about these resources. For example, when defining a device you provide physical information, such as its type and model, as well as logical information such as the identifier you will assign in the configuration definition.
- You must define an I/O configuration to the operating system (software) and the channel subsystem (hardware). The Hardware Configuration Definition (HCD) component of z/OS consolidates the hardware and software I/O configuration processes under a single interactive end-user interface.
- Processors** When more than one processor exists in a complex or more than one logical partition exists in a complex, z/OS is required to be defined in multisystem mode or a sysplex.
- Consoles** A console configuration consists of the various consoles that operators use to communicate with z/OS. Your installation first defines the I/O devices it can use as consoles with the hardware configuration definition (HCD). HCD manages the I/O configuration for the z/OS system. Once you have defined the devices, indicate to z/OS which devices to use as consoles by specifying the appropriate device numbers in the CONSOLxx parmlib member.

## 1.28 Choosing an install package

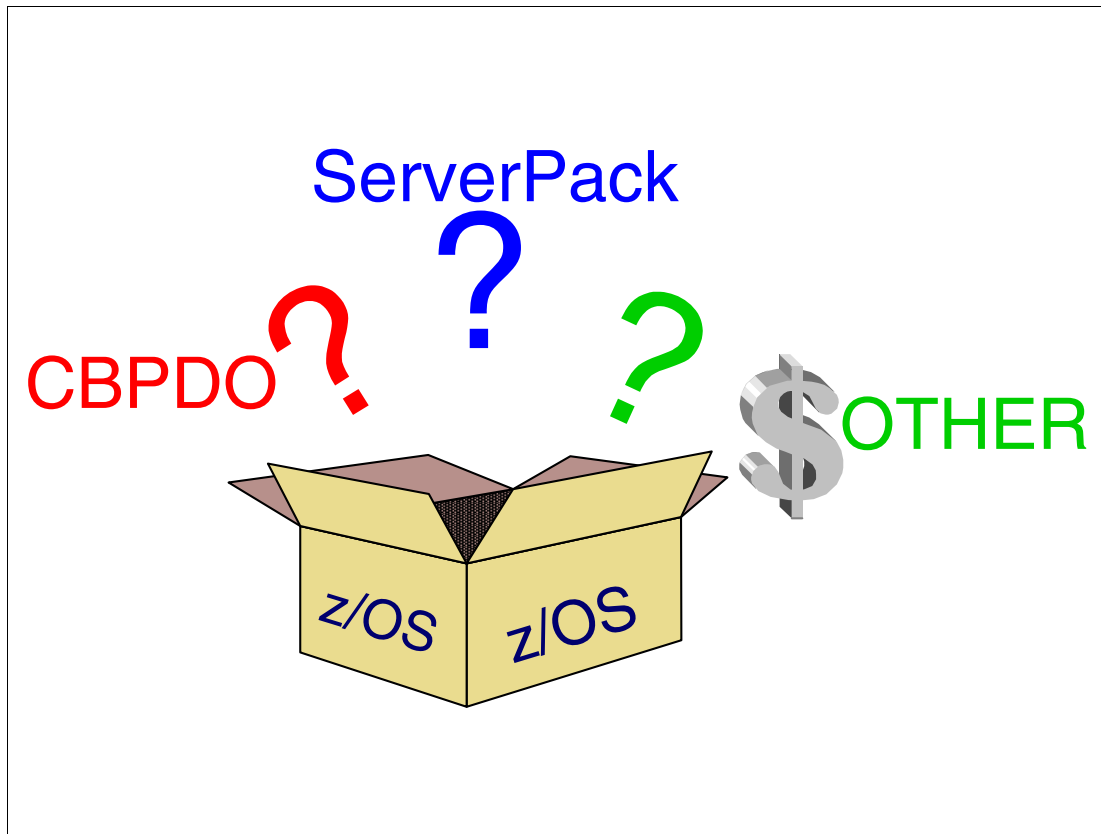


Figure 1-28 Installing z/OS

Because the base elements and optional features of z/OS are integrated into a single package with compatible service levels, you must install, with few exceptions, the entire z/OS product.

You can install z/OS using one of several IBM packages. Two of these packages are available at no additional charge when you license z/OS:

- ▶ ServerPac
- ▶ CBPDO

Other installation packages and offerings are available for a fee.

When you order a new system or a new release of z/OS, you also receive all the new maintenance or service that is applicable to the release.

The best installation method is usually the one that requires the least amount of work for you. We recommend the following:

- ▶ If you are new to z/OS and never had a previous system, use either a fee service or ServerPac's full system replacement option.
- ▶ If you're migrating from VM or VSE, use a fee service.
- ▶ If you're migrating from a level of products available before the general availability of MVS/ESA™ SP 4.3 (June of 1992), or if you're running unsupported levels of products, use a fee service or ServerPac's full system replacement.

- ▶ If you're migrating from MVS/ESA SP 4.3 or above, or any release of z/OS, use any method (ServerPac, CBPDO, or fee service).
- ▶ In response to customer feedback that the current release cycle is too short and complicates customer migration plans, the release schedule for z/OS and z/OS.e is changing from a six-month cycle to a 12-month cycle.
- ▶ New z/OS and z/OS.e functions will continue to be delivered between releases through the normal maintenance stream or as Web deliverables. In addition, significant new functions may be delivered between releases as features of the product.

## 1.29 Ordering z/OS

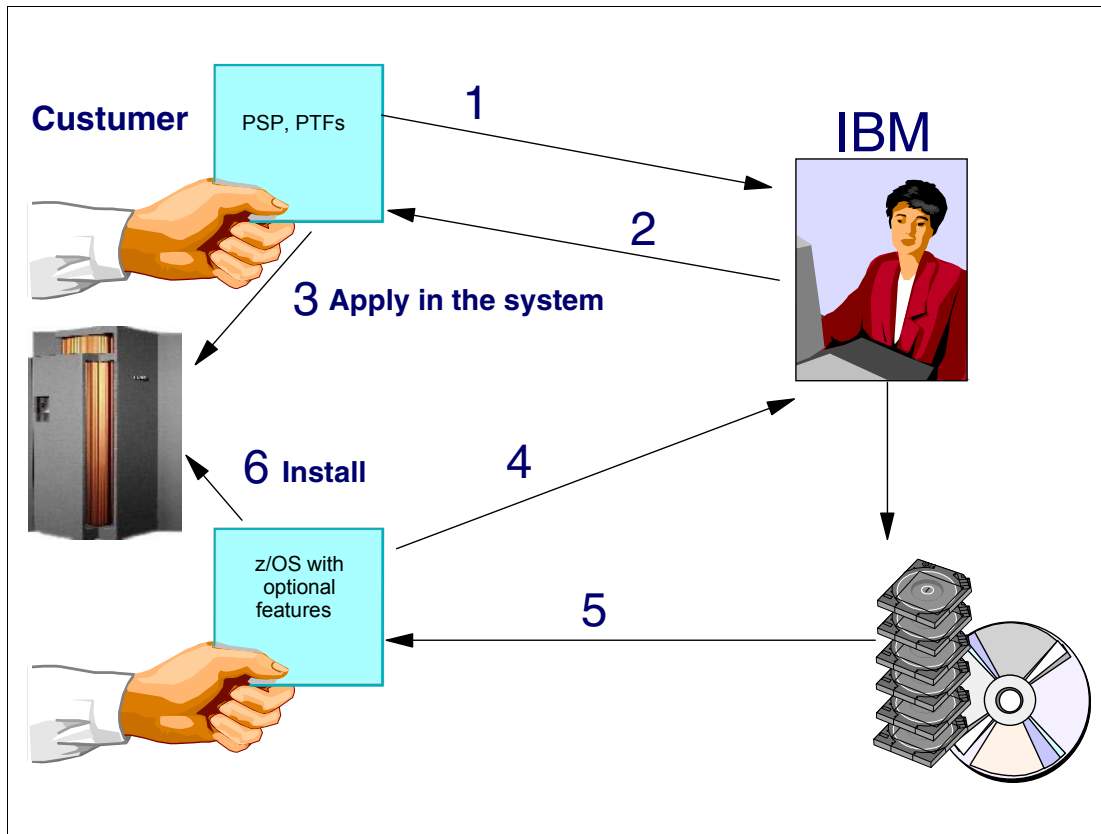


Figure 1-29 Products requiring customization

When ordering z/OS, be aware of there are two classifications of elements in the z/OS system:

1. Exclusive elements: Can be ordered only as part of the z/OS package, and are not available as independent elements or features anywhere else.
2. Non exclusive elements: Those elements or features included in the z/OS package that are also orderable as independent products, at the same functional level, from the z/OS product set.

When ordering products, be sure you include the following steps when planning your pre-installation activities:

- ▶ Obtain and install any required program temporary fixes (PTFs) or updated versions of the operating system.
- ▶ Call the IBM Software Support Center to obtain the preventive service planning (PSP) upgrade. This provides the most current information on PTFs. Have RETAIN® checked again just before testing products like RACF. Information for requesting the PSP upgrade can be found in the program directory. Although the program directory contains a list of the required PTFs, the most current information is available from the support center.
- ▶ Verify that your installation's programs will continue to run, and, if necessary, make changes to ensure compatibility with the new release.

## 1.30 Installing z/OS: ServerPac

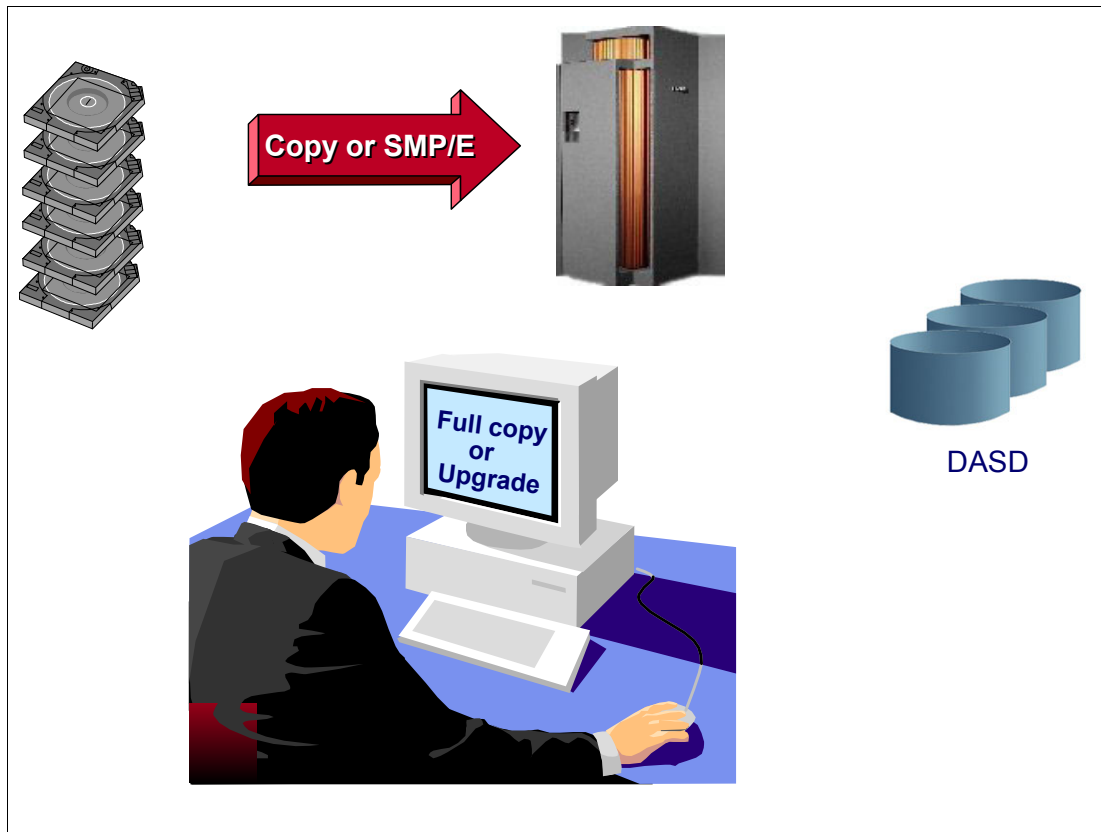


Figure 1-30 Installing z/OS: ServerPac

ServerPac is a software delivery package consisting of products and service for which IBM has performed the SMP/E installation steps and some of the post-SMP/E installation steps. To install the package on your system and complete the installation of the software it includes, you use the CustomPac Installation Dialog.

A z/OS ServerPac order contains an Interactive System Productivity Facility (ISPF) dialog that you use to install z/OS. This dialog is called the CustomPac Installation Dialog because it is used to install all of IBM's CustomPac offerings, for example, ServerPac, SystemPac®, FunctionPac, ProductPac®, and ServicePac®.

When ordering a ServerPac:

- ▶ IBM selects the products that were ordered.
- ▶ IBM integrates products and selected service into target and distribution libraries and their SMP/E zones.
- ▶ IBM enables the features that you ordered that use dynamic enablement.
- ▶ IBM verifies the resulting system for the specific package by doing an IPL, submitting a job, logging on to TSO/E, and checking the job's output. IBM performs this test using the operational data sets supplied with the ServerPac. If you use the software upgrade path when installing a ServerPac, you will use your own existing operational data sets, so this test will not assure that the system will IPL in your environment. However, it does make sure that the software itself can be used to IPL given a usable set of operational data sets.
- ▶ IBM selects all unintegrated service for products ordered and includes it on a service tape.

## 1.31 Installing z/OS: Custom-built product delivery option

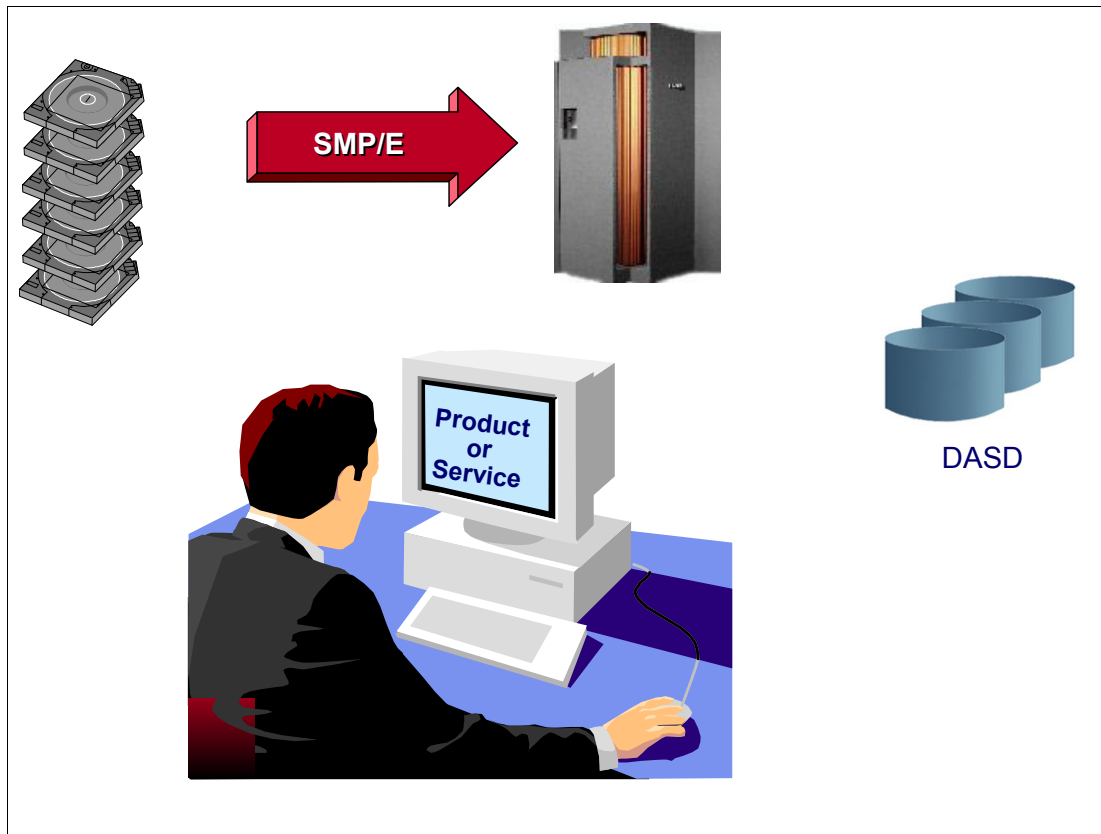


Figure 1-31 Installing z/OS: Custom-built product delivery option (CBPDO)

The custom-built product delivery option (CBPDO) is a software delivery package consisting of uninstalled products and unintegrated service. You must use SMP/E to install the individual z/OS elements and features, and their service, before you can IPL.

To order CBPDO, use an order checklist (available from an IBM representative, the z/OS Web site, or IBMLink™ via the configurator). The order is for a unique system release identifier (SREL). It is recommended that you order all products that you maintain in the same z/OS product set.

It is further recommended that you order only z/OS elements and features (or their equivalent stand-alone products) in your CBPDO order. (Ordering equivalent levels of non-exclusive elements does not increase the size of the CBPDO because the FMIDs are the same, but it does enable the IFAPRD00 PARMLIB member to be built correctly.) If you need to update other products, place a separate CBPDO order for these products. When ordering a CBPDO:

- ▶ IBM selects the products that were ordered.
- ▶ IBM selects service for the products you order and for products that are already licensed under the same customer number you use to place your order.
- ▶ IBM builds a customized job stream to enable the features that you ordered that use dynamic enablement.
- ▶ IBM builds a customized job stream to enable selected features that use the registration service.

## 1.32 Installing z/OS: Fee-based packages

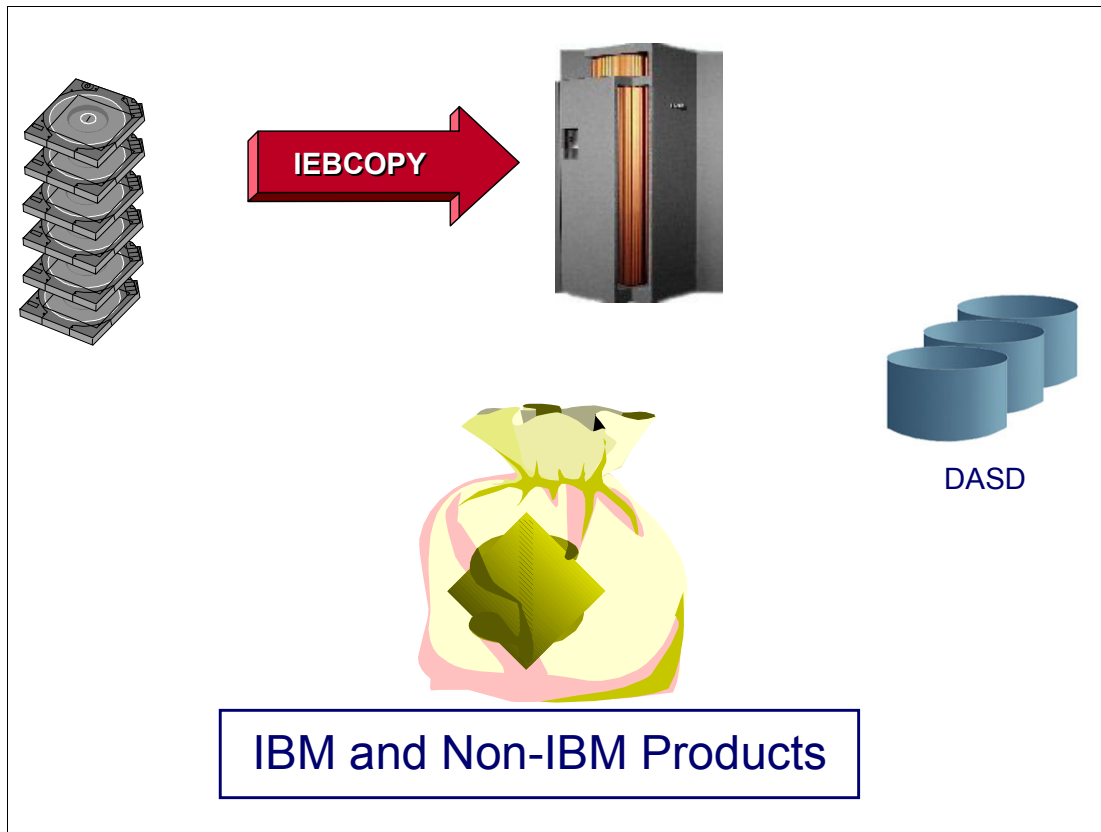


Figure 1-32 Installing z/OS using fee-based packages

Instead of using ServerPac or CBPDO to install z/OS, you could, for an additional fee, use one of the following services:

- ▶ SystemPac tailors z/OS to your environment (such as DASD layout, migration of MVSCP/IOCP to IODF, and naming conventions) based on information provided to IBM. With this offering, selected non-IBM products can be integrated. This offering can be delivered in IEBCOPY dump-by-data-set format or in full volume dump format.
- ▶ SoftwareXcel Installation Express (SIE), available in the U.S. only, provides prebuilt z/OS system packages in full volume dump format, tailored to customer hardware and software configurations. SIE includes on-site planning, installation, and package testing. SIE creates a compatibility research report for up to 70 non-IBM software products if requested. SIE can also include selected non-IBM software products integrated into the system package.
- ▶ The Entry Server Offering (available in some countries) is a packaged solution that includes hardware, software, installation services, maintenance, and financing to help customers get to current technology.
- ▶ Other fee-based help includes Washington System Center services, customized solutions, hardware services, and software services.





## z/OS storage concepts

This chapter describes some basic z/OS storage concepts, including the following:

- ▶ Virtual storage and address space concept
- ▶ Residence Mode and Addressing Mode
- ▶ How processor storage is managed by z/OS
- ▶ How virtual storage is managed by z/OS
- ▶ Address space map for 31-bit and 64-bits
- ▶ Dynamic address translation
- ▶ System address spaces
- ▶ Subsystem definitions
- ▶ Multiprogramming and multitask
- ▶ Module object and load module

The expressions *processor storage*, *central storage*, *main storage*, *real storage* or *memory* always refer to physical memory.

## 2.1 Processor storage overview

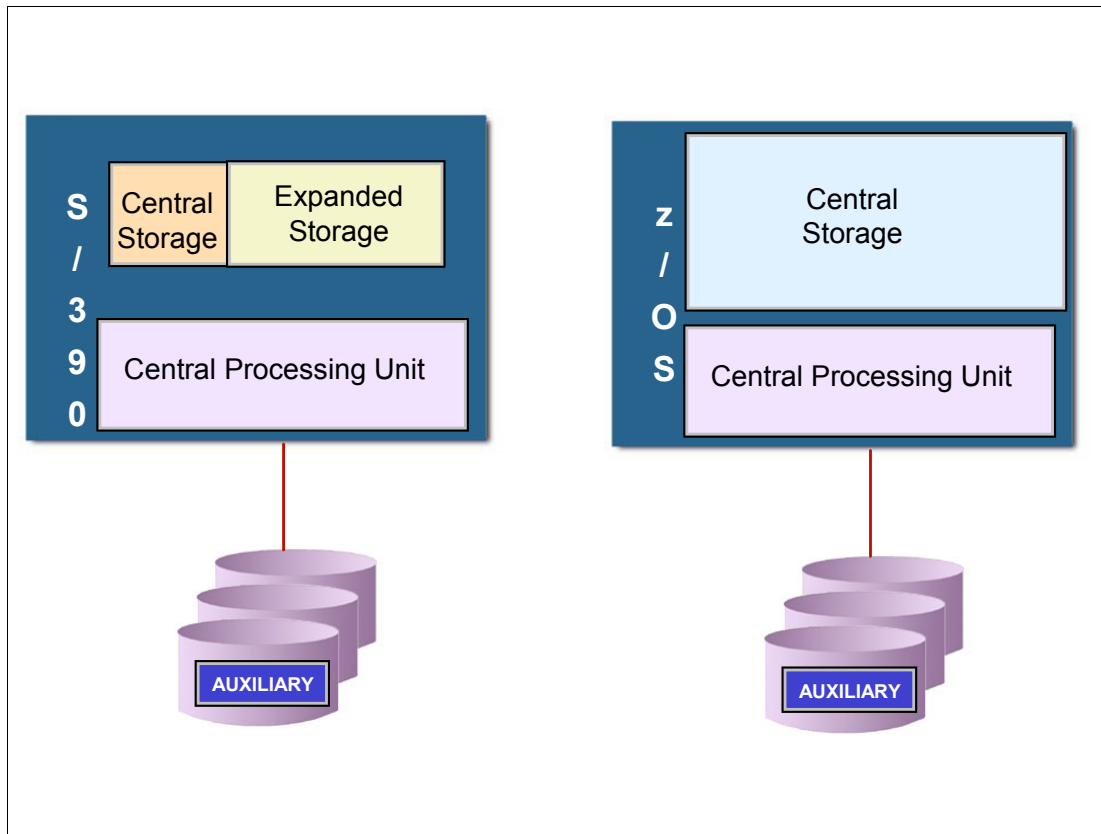


Figure 2-1 Overview of processor storage

From 370-XA until ESA/390 architecture, processor storage consisted of central plus expanded storage. In z/OS architecture *there is no* expanded storage.

The system initialization process begins when the system operator selects the LOAD function at the system console. MVS locates all of the usable central storage that is online and available to the system, and creates a virtual environment for the building of various system areas. The system uses a portion of both central storage and virtual storage.

- ▶ **Central Storage:** Central storage, also referred to as main storage, provides the system with directly addressable, fast-access electronic storage of data. Both data and programs must be loaded into central storage (from input devices) before they can be processed by the CPU. The maximum central storage size is restricted by hardware and system architecture as follows:
  - In the System/370™ architecture, the maximum main storage size is 16 megabytes.
  - From S/370™-XA architecture until ESA/390 architecture, the maximum memory size is 2 GB.
  - In z/Architecture the maximum central storage size is 16 exabytes.
- ▶ **Expanded Storage:** This is a form of electronic storage addressable in 4 KB blocks through the use of a 32-bit block number by special privileged instructions. The expanded storage was originally intended to bridge the gap in cost and density between main storage and magnetic media; later it provided a means to relieve the performance constraint imposed by the 31-bit real-address size by serving as a high-speed backing store for paging and for large data buffers.

- ▶ **CPU:** The central processing unit (CPU) is the controlling center of the system. It contains the sequencing and processing facilities for instruction execution, interruption action, timing functions, initial program loading and other machine-related functions.
- ▶ **Auxiliary Storage:** The auxiliary storage is in Direct Access Storage Devices (DASD) and is used to support basic system requirements as follows:
  - System data sets.
  - Paging data sets, which contain the paged-out portions of all virtual storage address spaces. In addition, output to virtual I/O devices may be stored in the paging data sets.

## 2.2 Virtual storage concepts

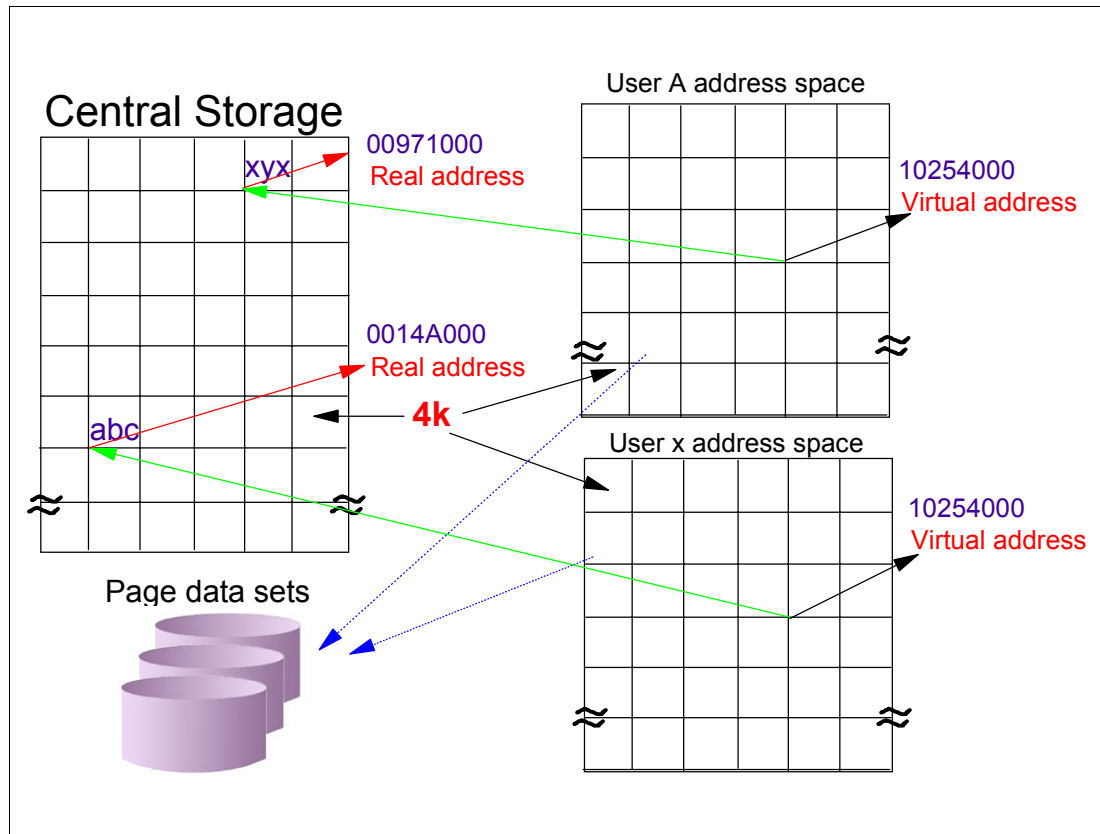


Figure 2-2 Virtual storage, auxiliary storage

Virtual storage is an illusion created by the architecture, in that the system seems to have more memory that it really has. The virtual storage concept was introduced in the Atlas System used for the first time in the IBM System/370 architecture. It is based on:

- ▶ An address is an identifier of a required piece of information, but not a description of where in main memory that piece of information is.<sup>1</sup> This allows the size of an address space (all addresses available to a program) to exceed the main storage size.
- ▶ All main storage references are made in terms of virtual storage address.
- ▶ A hardware mechanism is employed to perform a mapping between the virtual storage address and its physical location.
- ▶ When a requested address is not in main storage, an interruption is signaled and the required data is brought into memory.

In an MVS operating system, virtual storage was implemented using:

- ▶ Pages: The main storage and program address space are divided into fixed blocks of 4K bytes in size, implemented in a manner that is transparent to the user.
- ▶ Segment: Program address space is divided into segments of 1M bytes in size.
- ▶ Dynamic address translation, which is implemented by hardware and by software throughout page tables and segment tables.

A z/OS program resides in virtual storage and only parts of the program currently active need to be in real storage at processing time. The inactive parts are held in auxiliary storage.

<sup>1</sup> J.Fotheringham, *Dynamic storage allocation in the Atlas computer, including the use of a baking storage*. Communications of the ACM (Nov 1961)

## 2.3 Frames, slots, and pages

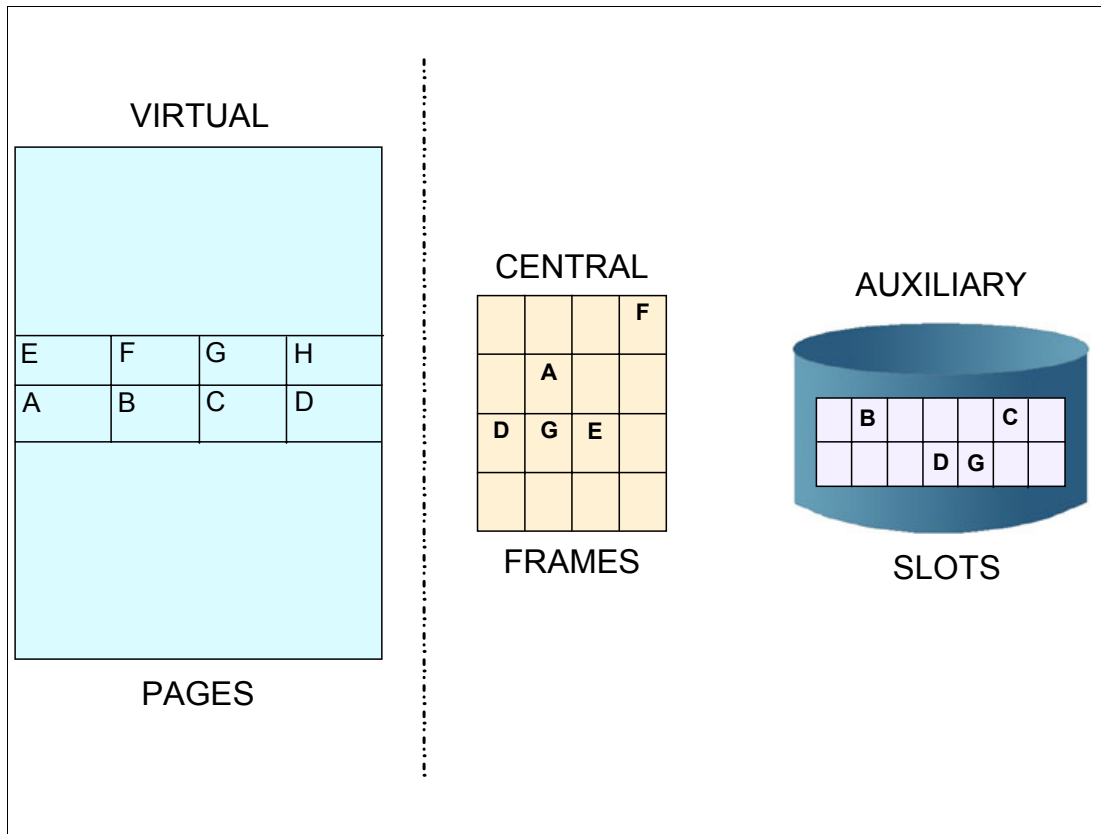


Figure 2-3 Storage frames, slots, and pages

When a program is selected, the system brings it into virtual storage and divides it into pages of 4K bytes. The system transfers the pages of a program into central (real) storage for execution and out to auxiliary storage when not needed. To the programmer the entire program appears to occupy contiguous space in central storage at all times. Actually, not all pages of a program are necessarily in central storage at one time. Also, the pages that are in central storage do not necessarily occupy contiguous space.

The parts of a program executing in virtual storage must be moved between real and auxiliary storage. To allow this, z/OS breaks the storage into blocks of 4K:

- ▶ A block of real storage is a *frame*.
- ▶ A block of virtual storage is a *page*. The virtual storage is backed by:
  - Central storage
  - Auxiliary storage
- ▶ A block of storage on an auxiliary device is a *slot*.

Frames, pages, and slots are all the same size, 4 KB.

## 2.4 The address space concept

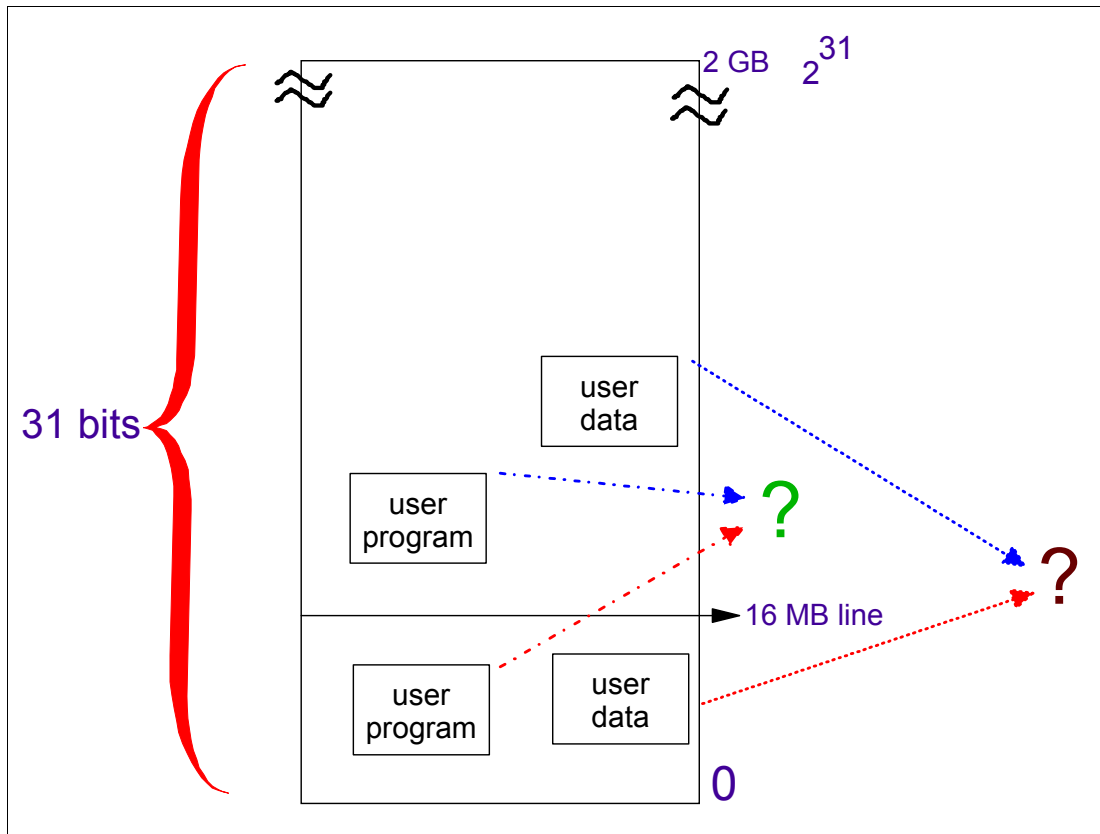


Figure 2-4 Address space concept

The System/370 was the first IBM architecture to use virtual memory and address space concepts. The address space is an area of contiguous virtual addresses available to a program and its data. The range of virtual addresses available to a program starts at 0 and can go to the highest address permitted by the operating system architecture. This virtual storage is available for user code and data. Because it maps all of the available addresses, an address space includes system code and data as well as user code and data. Thus, not all of the mapped addresses are available for user code and data.

The S/370 architecture used 24 bits for addressing (3 bytes). So, the highest accessible address in the MVS/370 was 16 megabytes, which was also the address space size.

With the MVS Extended Architecture, S/370-XA, the system extended to 31 bits for addressing (4 bytes) and the highest address and the address space size went from 16 megabytes to 2 gigabytes, 128 times bigger. The 16 MB address became the division point between the two architectures and is commonly called the *line*.

For compatibility, programs running in MVS/370 should run in MVS/370-XA, and new programs should be able to exploit the new technology. So, the high order bit of the address (4 bytes) is *not* used for addressing, but rather to indicate how many bits are used to solve an address: 31 bits (bit 32 on) or 24 bits (bit 32 off).

## 2.5 Addressing mode and residence mode

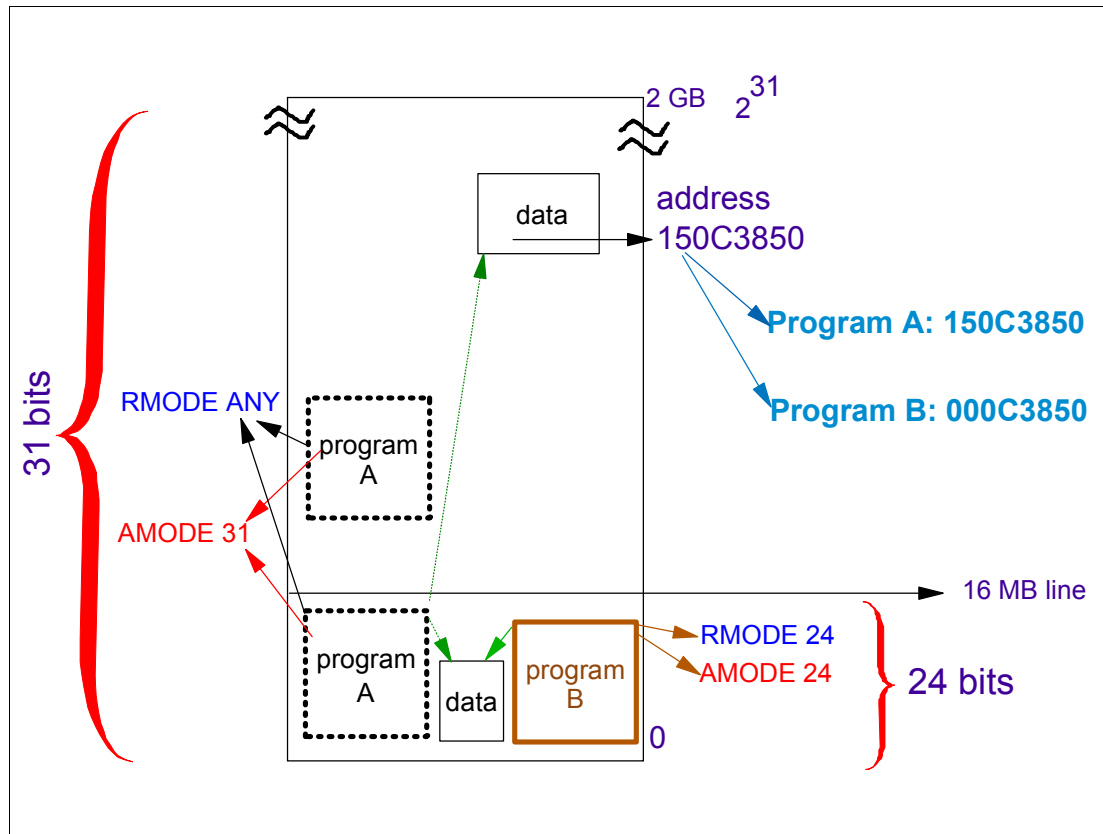


Figure 2-5 Addressing mode and residence mode

With the MVS/370-XA came the concept of *addressing mode* (AMODE), a program attribute to indicate which hardware addressing mode should be active to solve an address, that is, how many bits should be used for solving addresses.

With MVS/370-XA also came the concept of *residence mode* (RMODE), to indicate where a program should be placed in the virtual storage, when the system loads it from DASD:

- ▶ RMODE=24: Indicates that the module must reside below the 16-MB virtual storage line.
- ▶ RMODE= ANY: Indicates that the module might reside anywhere in virtual storage either above or below the 16-MB virtual storage line.

AMODE and RMODE are load module attributes and are placed in the load module's directory entry in the partitioned data set.

## 2.6 Storage managers



Figure 2-6 The storage component managers

In a z/OS system, storage is managed by the following storage components managers:

- ▶ **Real Storage Manager (RSM):** Controls the allocation of central storage during system initialization, and pages in user or system functions during execution. Some specific RSM functions are:
  - Allocate central storage to satisfy GETMAIN requests for SQA and LSQA.
  - Allocate central storage for page fixing.
  - Allocate central storage for an address space that is to be swapped in.
  - Allocate and initialize control blocks and queues related to expanded storage.<sup>2</sup>
- ▶ **Virtual Storage Manager (VSM):** Each installation can use virtual storage parameters to specify how certain virtual storage areas are to be allocated. These parameters have an impact on central storage use and overall system performance.
- ▶ **Auxiliary Storage Manager (ASM):** The auxiliary storage manager code controls the use of page and swap data sets. As a system programmer, you are responsible for:
  - Page and swap operations
  - Page and swap data set sizes
  - Space calculation
  - Performance of page and swap data sets
  - Estimating the total size of the paging data sets

<sup>2</sup> Expanded storage does not exist in a z/OS environment.



## 2.7 Virtual storage manager

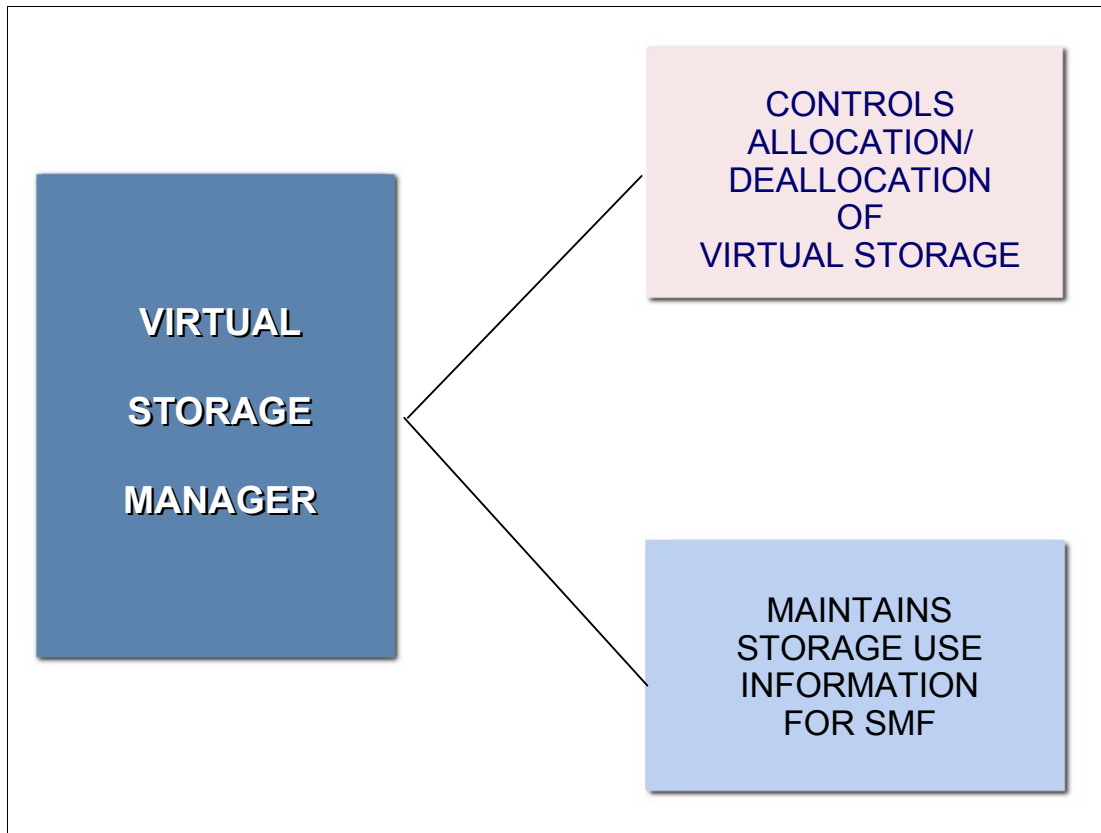


Figure 2-7 Virtual storage manager

Virtual storage is managed by the virtual storage manager (VSM); VSM's main function is to control the use of virtual storage addresses. The management of virtual storage allows you to write large programs without the need for complex overlay structures.

Virtual storage is requested through the use of the GETMAIN or STORAGE OBTAIN macro and returned to the virtual storage manager using the FREEMAIN or STORAGE RELEASE macro.

The VSM functions are:

- ▶ Allocate and release blocks of virtual storage on request.
- ▶ Ensure that real frames exist for SQA, LSQA, and V=R pages.
- ▶ Associate a storage protection key with each virtual storage block requested. This function is provided through the GETMAIN and STORAGE macros.

In addition, the VSM provides additional services through the use of the following macros:

- ▶ VSMREGN macro: List the starting address and the size of the private area regions associated with a given task.
- ▶ VSMLOC macro: Verify that a given area has been allocated through a GETMAIN or STORAGE macro.
- ▶ VSMLIST macro: List the ranges of virtual storage allocated in a specified area.

These system services are especially useful when determining available storage, coding recovery procedures, or specifying areas to be included in a dump. VSMREGN enables you to determine the amount of storage that you have for potential use.

## 2.8 Real storage manager

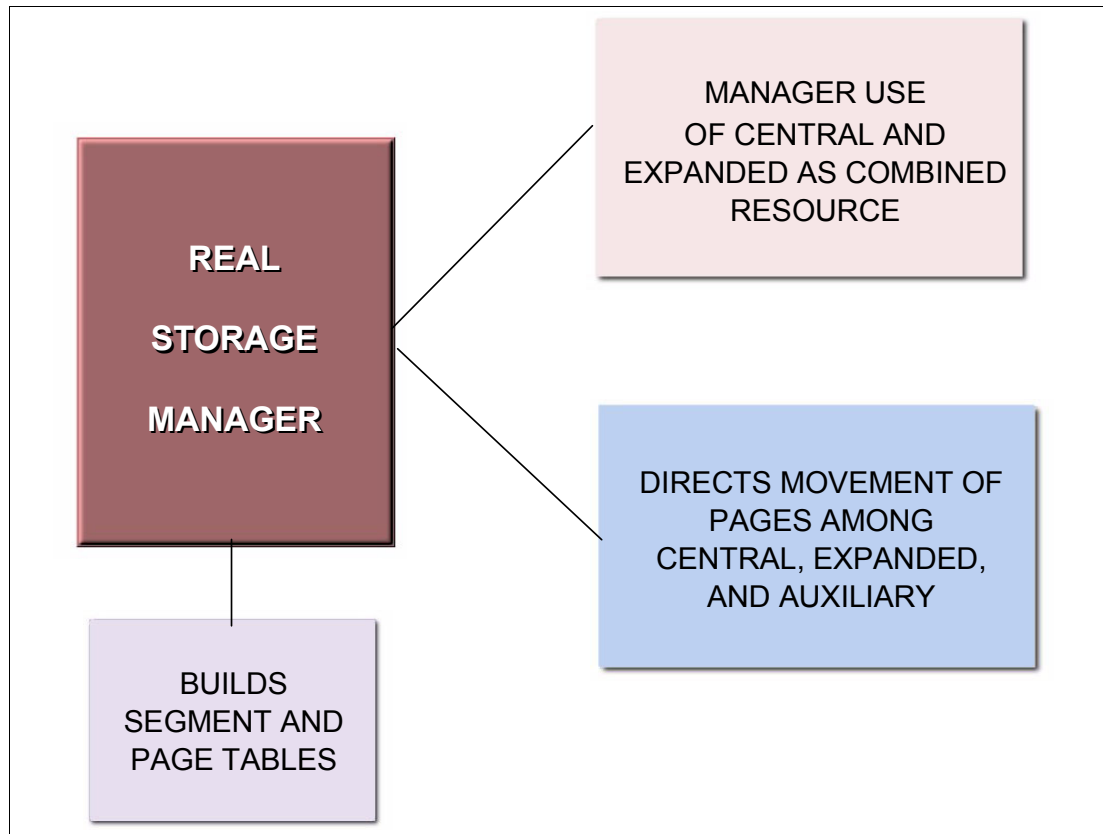


Figure 2-8 Real storage manager

The task of the real storage manager (RSM) is to control the usage of real storage frames. RSM acts together with the auxiliary storage manager (ASM) to support the virtual storage concept, and with VSM to ensure that a GETMAINed page is backed up in a real storage frame. Furthermore, RSM establishes many services to other components and application programs to manipulate the status of pages and frames.

RSM controls the allocation of central storage during initialization, and pages in user or system functions for execution. Some RSM functions are to:

- ▶ Allocate central storage to satisfy GETMAIN requests for SQA and LSQA
- ▶ Allocate central storage for page fixing
- ▶ Allocate central storage for an address space that is to be swapped in
- ▶ Initialize control blocks and queues related to expanded storage<sup>3</sup>

### Expanded storage

Expanded storage can be thought of as an expansion of central storage. The purpose of expanded storage is to reduce the paging and swapping of pages between central storage and auxiliary storage, and thus enhance system performance. Because moving a page between central storage and expanded storage is much faster than I/O, use of expanded storage can provide a significant performance advantage. In z/Architecture, with the possibility of huge central storage, expanded storage is no longer supported.

<sup>3</sup> Expanded storage is no longer supported in z/Architecture.

## 2.9 Auxiliary storage manager

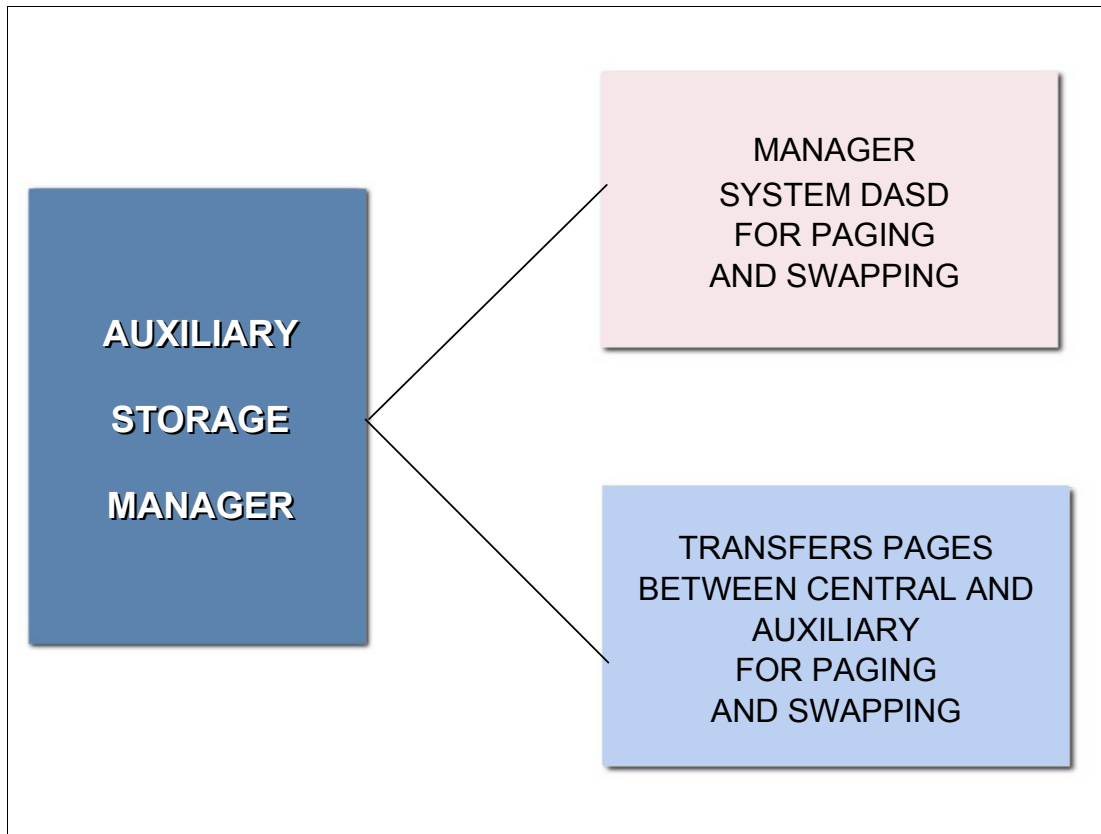


Figure 2-9 Auxiliary storage manager

Auxiliary storage manager (ASM) is the component of the z/OS system responsible for transferring virtual pages between real and auxiliary storage. This is done as either a paging operation (one page at a time) or as a swapping operation (an address space at a time). ASM manages the transfer by initiating the I/O and by maintaining tables to reflect the current status of auxiliary storage.

To page efficiently and expediently, ASM divides the pages of the system into classes, namely PLPA, common, and local. There is at least one page data set for each class. In addition, output to virtual I/O devices may be stored in local paging data sets. Page data sets are created by the system programmer through the **DEFINE IDCAMS** command.

The paging controllers of the auxiliary storage manager attempt to maximize I/O efficiency by incorporating a set of algorithms to distribute the I/O load as evenly as is practical. In addition, every effort is made to keep the system operable in situations where a shortage of a specific type of page space exists.

ASM selects a local page data set for page-out from its available page data sets. ASM selects these data sets in a circular order within each type of data set, subject to the availability of free space and the device response time.

If the address space is swapped directly from central storage to auxiliary storage, ASM reads and writes these working set pages in parallel.

## 2.10 Paging and swapping

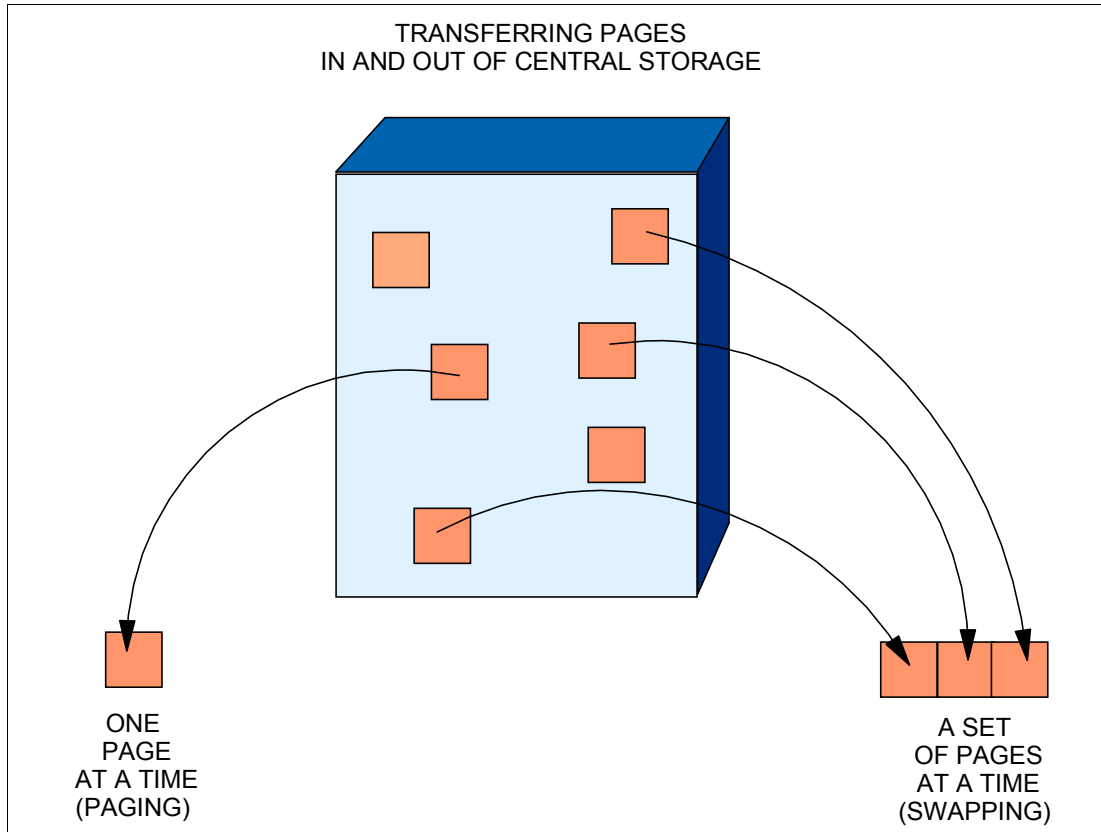


Figure 2-10 Paging and swapping concepts

The paging and swapping controllers of the auxiliary storage manager attempt to maximize I/O efficiency by incorporating a set of algorithms to distribute the I/O load as evenly as is practical. In addition, every effort is made to keep the system operable in situations where a shortage of a specific type of page space exists.

In ESA390 architecture, RSM uses expanded storage as an extension of central storage. When a page is to be removed from central storage, RSM first considers moving it to expanded storage instead of auxiliary storage. When a page that is needed is not in central storage, RSM first checks expanded storage for the page. If the page is in expanded storage, RSM synchronously retrieves the page. If the page is not in expanded storage, RSM calls ASM to schedule asynchronously the paging I/O to retrieve the page from auxiliary storage. When contention for expanded storage increases, the system removes pages from expanded storage to free expanded storage frames. RSM first moves the pages from expanded storage to central storage. RSM then calls ASM to schedule the paging I/O necessary to send these pages to auxiliary storage. This process is called migration. Migration completes when the pages are actually sent to auxiliary storage.

RSM is responsible for reclaiming the central storage allocated to an address space when the address space is to be swapped out of central storage. RSM is also responsible for building the control structures necessary to efficiently swap the address space back into central storage.

## 2.11 Auxiliary data sets

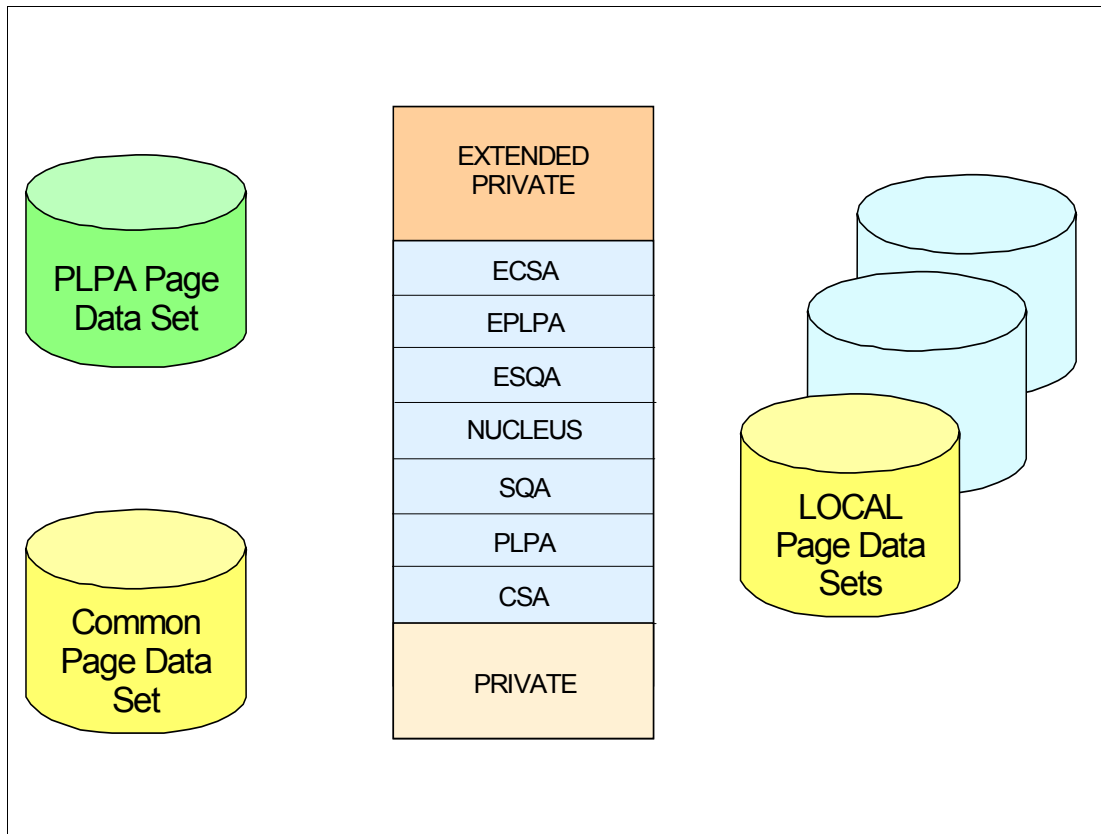


Figure 2-11 Paging to auxiliary data sets

### Paging

To page efficiently and expediently, ASM divides the pages of the system into classes, namely PLPA, common, and local. Contention is reduced when these classes of pages are placed on different physical devices. Although the system requires only one local page data set, performance improvement can be obtained when local page data sets are distributed on more than one device, even though some devices may be large enough to hold the entire amount of necessary page space. The PLPA and common page data sets are both required data sets, and there can be only one of each. Spillage back and forth between the PLPA and common page data sets is permissible, but, in the interest of performance, only spilling from PLPA to common should be tolerated.

The page data sets are:

- ▶ PLPA page data set: This contains pageable link pack area. (There is only one.)
- ▶ Common page data set: This contains the non-PLPA virtual pages of the system common area. (There is only one.)
- ▶ Local page data set: This contains the private area (address space) pages, and space for any VIO data sets. Local page data sets can be dynamically added to and deleted from the paging configuration without re-IPLING the system. (There are one or more.)
- ▶ Duplex page data set: This is an optional data set, used when duplexing of the common area is requested.

## 2.12 31-bits address space map

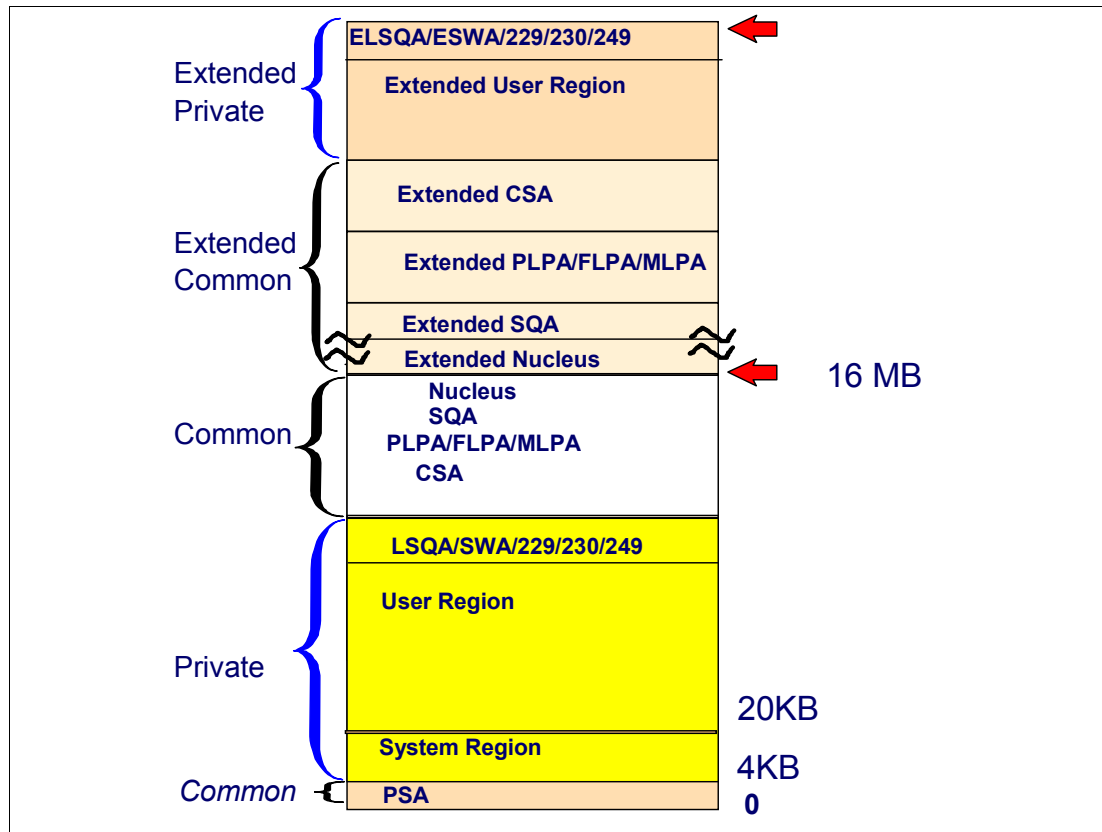


Figure 2-12 31-bits address space map

Since the introduction of the MVS/370-XA architecture, the address space size is 2 GB because the system immediately started using the area above the line.

The virtual address space is divided in areas according to the purpose of their use. Virtual storage allocated in each address space is divided between the system's requirements and the user's requirements. The base system control programs require space from each of the basic areas. Each virtual address space consists of:

- ▶ The common area below 16 MB
- ▶ The private area below 16 MB
- ▶ The extended common area above 16 MB
- ▶ The extended private area above 16 MB

Most of the system areas exist both below and above 16 megabytes, providing an environment that can support both 24-bit and 31-bit addressing. However, each area and its counterpart above 16 megabytes can be thought of as a single logical area in virtual storage.

## 2.13 The common virtual storage area

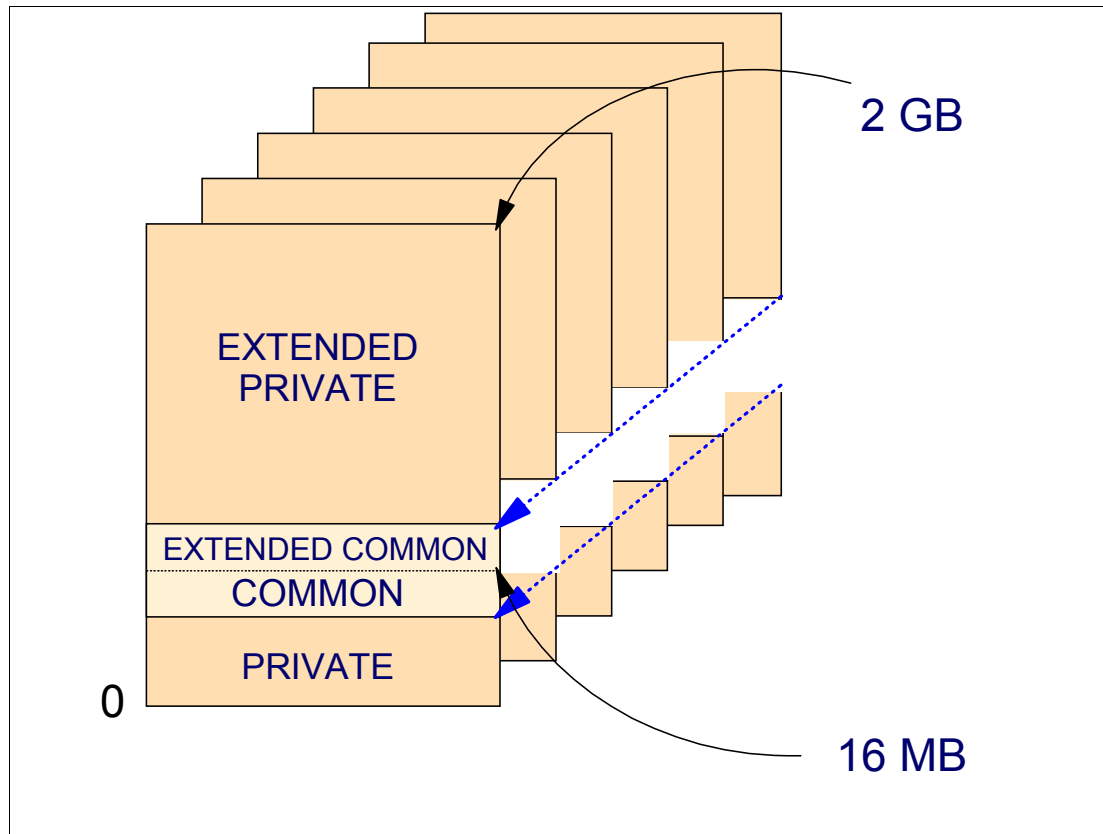


Figure 2-13 Common storage area

All address spaces in a z/OS system image share a virtual storage area. That means that all address spaces in that system image access the same data in the same virtual address. The following storage areas are located in the common area:

- ▶ Prefixed storage area (PSA)
- ▶ Common service area (CSA)
- ▶ Pageable link pack area (PLPA)
- ▶ Fixed link pack area (FLPA)
- ▶ Modified link pack area (MLPA)
- ▶ System queue area (SQA)
- ▶ Nucleus

Each storage area in the common area (below 16 MB) has a counterpart in the extended common area (above 16 MB) with the exception of the PSA.

The PSA size is fixed by the system architecture. The PSA size is 4K in s/370, 370-XA, and ESA/390 architectures, and 8K in z/architecture.

The CSA and SQA sizes are settled during the IPL, accordingly to system initialization parameters in the SYS1.PARMLIB(IEASYSxx) system data set.

## 2.14 z/OS nucleus

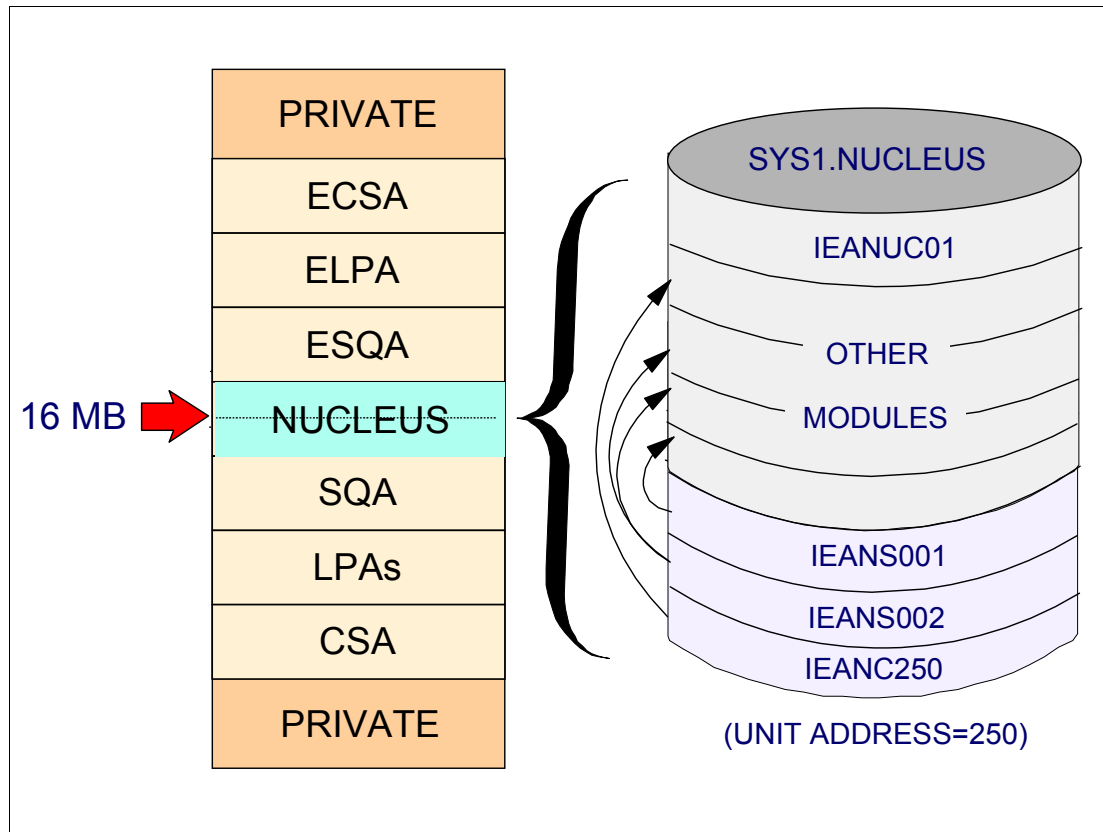


Figure 2-14 z/OS nucleus

The nucleus area contains the nucleus load module and extensions to the nucleus that are initialized during IPL processing. The nucleus includes a base and an architectural extension.

The modules to be added to the nucleus region, or deleted from it, must reside in members of SYS1.NUCLEUS. The system programmer can add or delete modules from the nucleus by simply specifying the members on INCLUDE or EXCLUDE statements in SYS1.PARMLIB(NUCLSTxx).

The nucleus is always fixed in central storage.



## 2.15 System queue area (SQA/ESQA)

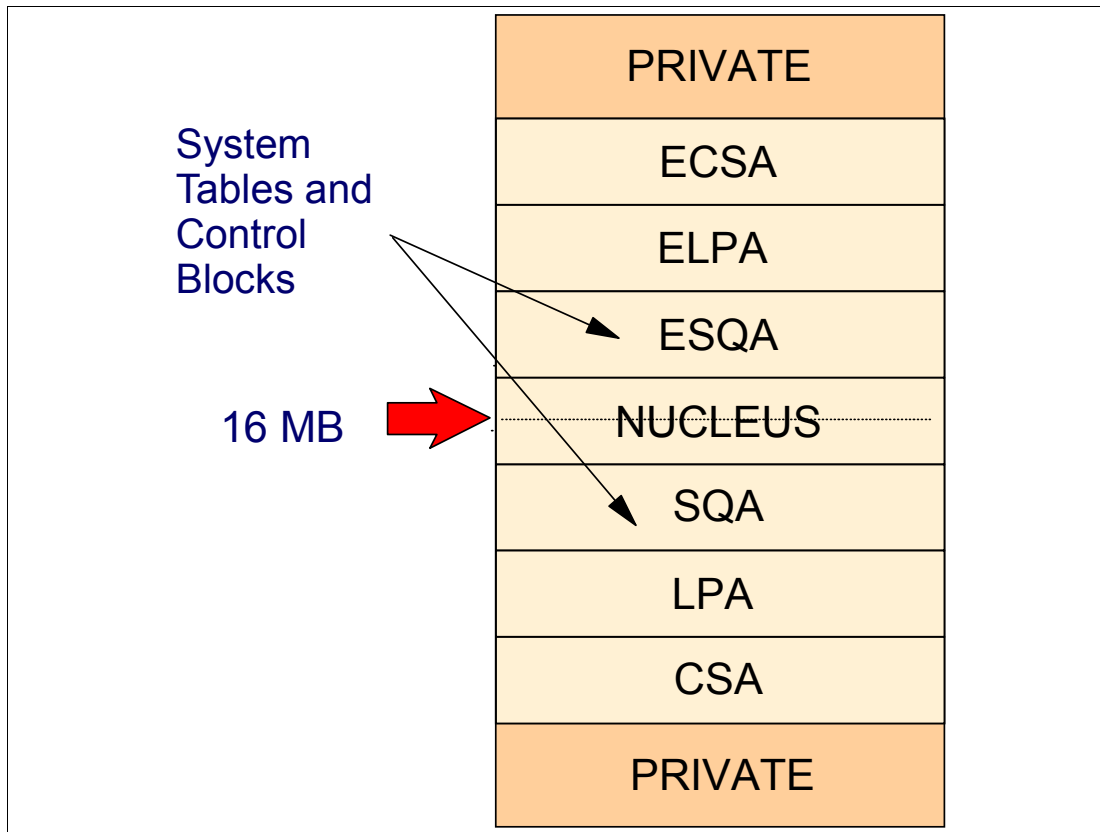


Figure 2-15 System queue area (SQA and ESQA)

The system queue area (SQA) contains tables and queues relating to the entire system. Its contents are highly dependent on configuration and job requirements at an installation. The total amount of virtual storage and the number of private virtual storage address spaces are two of the factors that affect the system's use of SQA.

The SQA is allocated directly below the nucleus; the extended SQA (ESQA) is allocated directly above the extended nucleus. The size of the SQA can be specified through the:

- ▶ SQA parameter in the IEASYSxx member of SYS1.PARMLIB
- ▶ Nucleus Initialization Process (NIP) or operator's console

During the initialization process, if the SQA required by the system configuration exceeds the amount that has been reserved through the SQA parameter, the system attempts to allocate additional virtual SQA from the CSA area. If less than eight frames, below 16 MB, remain for allocation purposes, the system stops creating new address spaces. When SQA is in use, it is fixed in central storage. After the initialization process, all SQA requirements are allocated in 4K frames as needed. These frames are placed within the preferred area (above 16 MB, if possible) to keep long-term resident pages grouped together.

Ensuring the appropriate size of ESQA and ECSA is critical to the long-term operation of the system. If the size allocated for ESQA is too small or is used up very quickly, the system attempts to use ECSA. When both ESQA and ECSA are used up, the system allocates space from SQA and CSA below 16 megabytes. The allocation of this storage could eventually lead to a system failure.

## 2.16 Common service area (CSA and Extended CSA)

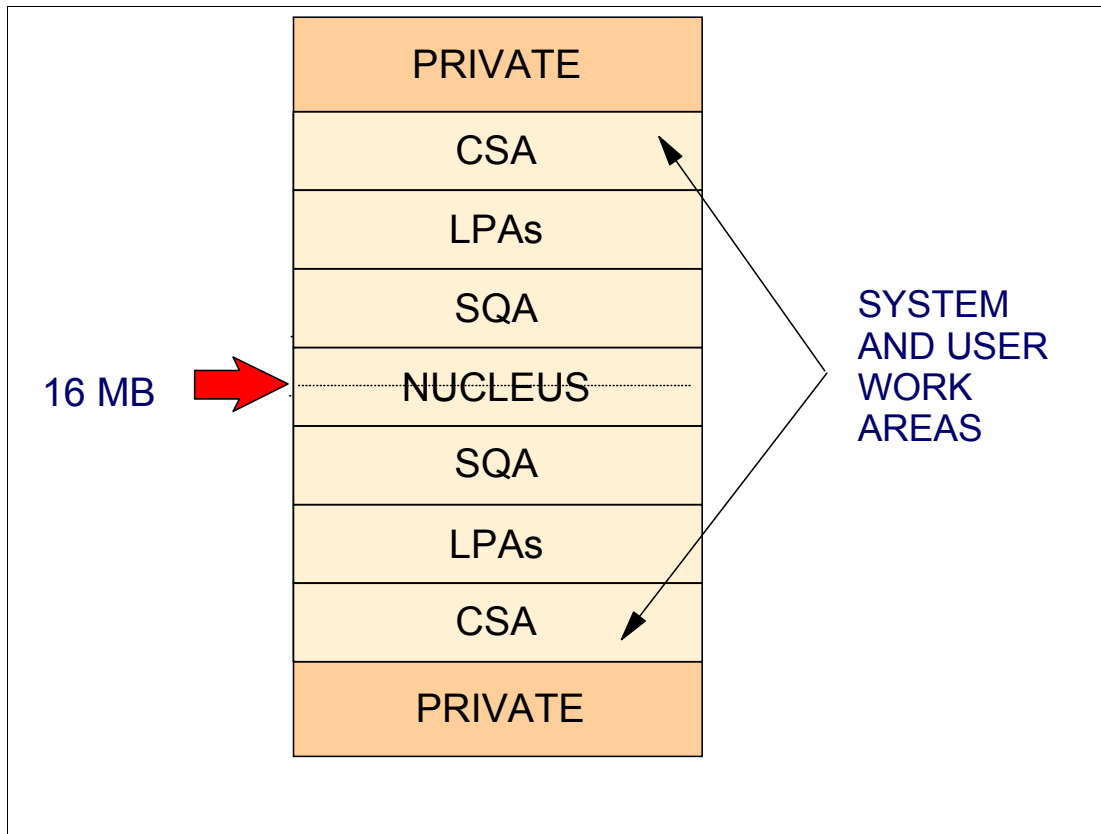


Figure 2-16 Common service area (CSA and ECSA)

The common service area (CSA) contains pageable and fixed data areas that are addressable by all active virtual storage address spaces. CSA normally contains data referenced by a number of system address spaces, enabling address spaces to communicate by referencing the same piece of CSA data.

CSA is allocated directly below the MLPA; extended CSA is allocated directly above the extended MLPA. If the virtual SQA space is done, the system allocates additional SQA space from the CSA. The size of the CSA is specified by:

- ▶ The CSA parameter in the IEASYSxx member of SYS1.PARMLIB.
- ▶ The CSA parameter at the operator's console during system initialization. This value overrides the current system parameter value for CSA that was established by IEASYSxx.

**Note:** If the size allocated for extended SQA is too small or is used up very quickly, the system attempts to steal space from extended CSA. When both extended SQA and extended CSA are used up, the system allocates space from SQA and CSA below 16 MB. The allocation of this storage could eventually lead to a system failure. Ensuring the appropriate size of extended SQA and extended CSA storage is critical to the long-term operation of the system.

## 2.17 Link pack area (LPA and Extended LPA)

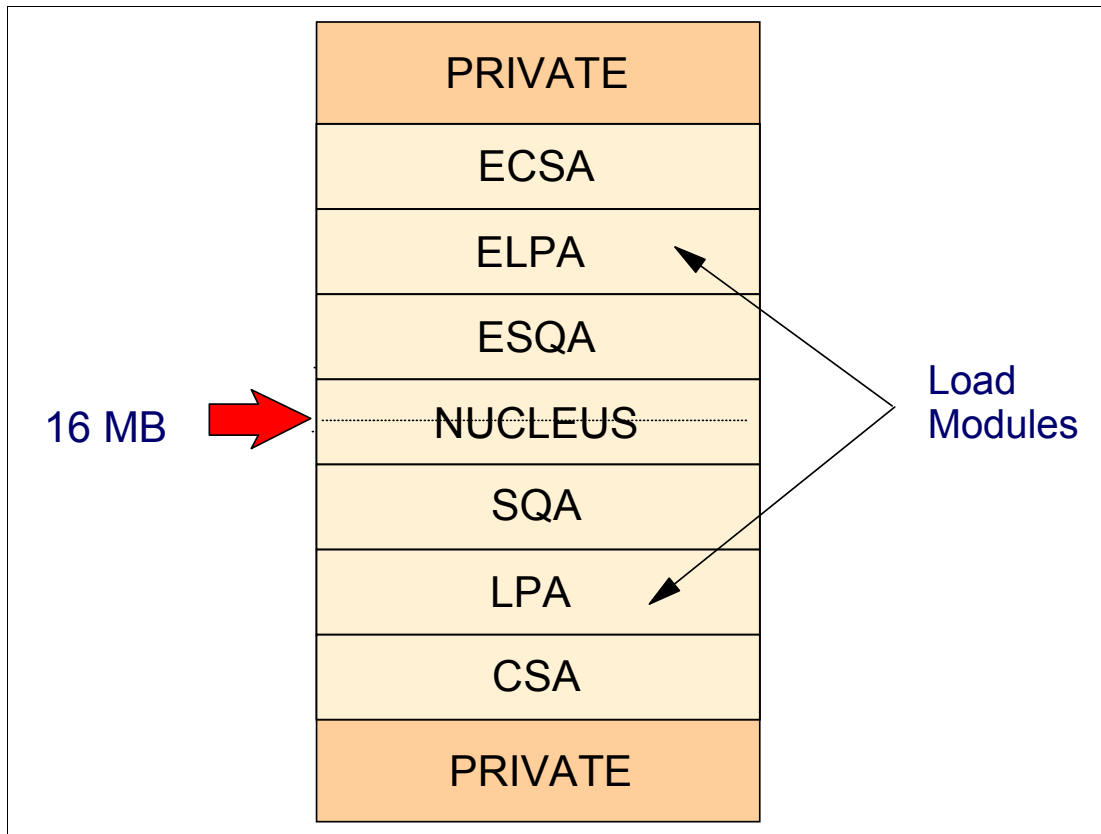


Figure 2-17 Link pack area (LPA and ELPA)

The link pack area (LPA) contains SVC routines, access methods, and other read-only system programs along with any read-only reentrant user programs, selected by an installation, which can be shared among users of the system. Because these modules are reentrant, their copy is not self-modifying, meaning that the same copy of the module can be used by any number of tasks in any number of address spaces at the same time, thus reducing the demand for central storage and the program fetch overhead.

The LPA size depends on the number of modules loaded in it. The LPA boundaries are in megabyte segments. When modules are added to LPA, the growth in LPA can cause the common area to cross one or more segment boundaries, which reduces the available private area by a corresponding amount; each time the common area crosses a segment boundary, the available private area for all address spaces in the system is reduced by the size of one segment, even for those address spaces not using the load modules added to LPA.

The extended link pack area (ELPA) is built above 16 megabytes. The only difference is that common storage areas above 16 megabytes are built from 16 megabytes upward, while those below 16 megabytes are built from 16 megabytes downward.

All modules placed in LPA are assumed to be APF-authorized. Being APF authorized means that a program can invoke any SVC routine accessing protected system and private areas. All IBM publications identify libraries that can be placed in the LPA list safely, and many list modules you should consider placing in LPA to improve the performance of specific subsystems and applications.

The LPA is divided into:

- ▶ Pageable LPA (PLPA/EPLPA)
- ▶ Fixed LPA (FLPA/EFLPA): This area is used for modules that have to be fixed in memory, instead of pageable. The modules to be fixed are specified in system initialization parameters or through the operator's console at system initialization.
- ▶ Modified LPA (MLPA and EMLPA): The MLPA can be used at IPL time to temporarily modify or update the PLPA with new or replacement modules.

Modules placed anywhere in LPA are always in virtual storage, and modules placed in FLPA are also always in central storage. Whether modules in LPA, but outside FLPA, are in central storage depends on how often they are used by all the users of the system, and on how much central storage is available. The more often an LPA module is used, and the more central storage is available on the system, the more likely it is that the pages containing the copy of the module will be in central storage at any given time.

Each address space uses the same common area. Portions of the common area are paged in and out as the demands of the system change and as new user jobs (batch or time-shared) start and old ones terminate.

## 2.18 31-bit private area

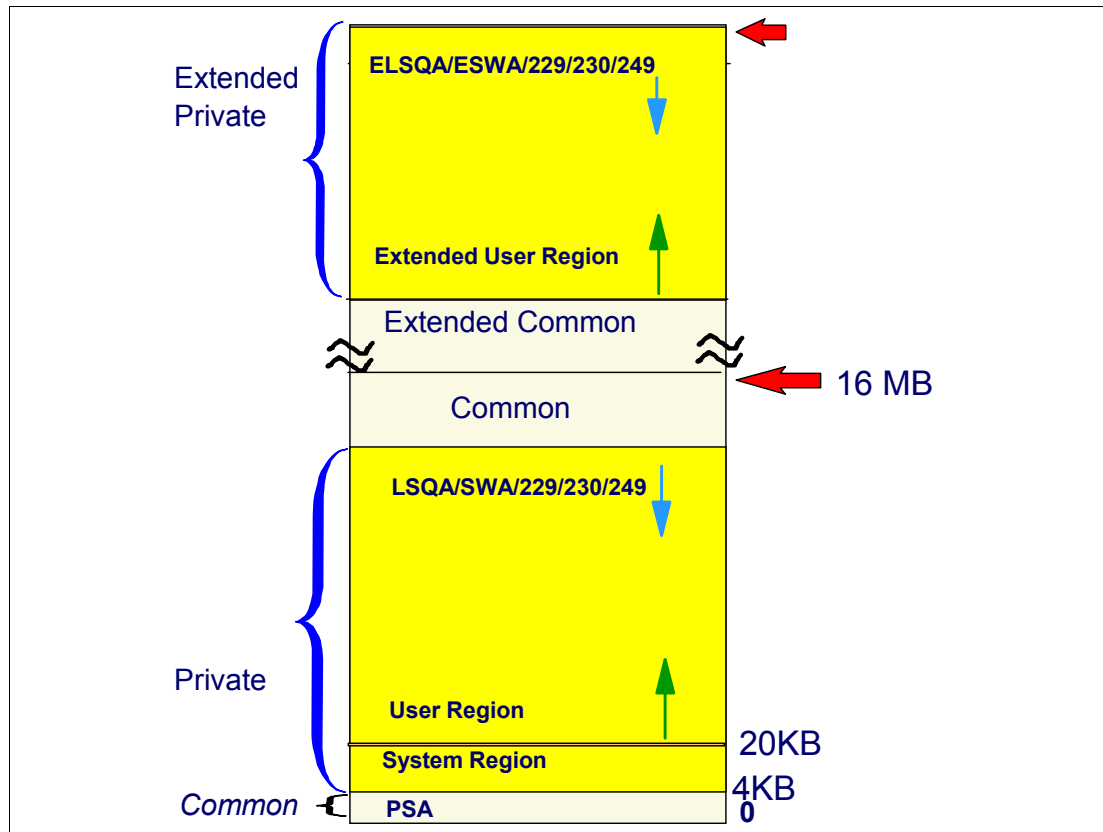


Figure 2-18 The user's private area (user region)

The private area contains:

- ▶ **Subpools 229, 230, and 249:** This area allows local storage to be obtained in the requestor's storage protect key. The area is used for control blocks that can be obtained only by authorized programs having appropriate storage protect keys. These control blocks were placed in storage by system components on behalf of the user.
- ▶ **Local System Queue Area (LSQA):** Contains tables and queues associated with the user's address space. LSQA is intermixed with SWA and subpools 229, 230, and 249 downward from the bottom of the CSA into the unallocated portion of the private area, as needed. ELSQA is also intermixed but is allocated downward from 2G into the unallocated portion of the extended private area, as needed. LSQA does not take space below the top of the highest storage currently allocated to the user region.
- ▶ **Scheduler Work Area (SWA):** This area contains control blocks that exist from task initiation to task termination. It includes control blocks and tables created during JCL interpretation.
- ▶ **A 16 KB system region area.**
- ▶ **User region for running user program applications and storing user data.**

When a module is loaded into the private area for an address space, the region available for other things is reduced by the amount of storage used for the module. The amount of private storage a job can use below 2G is limited through the REGION keyword on the JOB or EXEC Job Control Language (JCL) statements. Also, the region size can be controlled and overridden through the SMF exit IEFUSI. A value equal to 0K or 0M gives the job all private storage available below 2G.

## 2.19 Data spaces and hiperspace

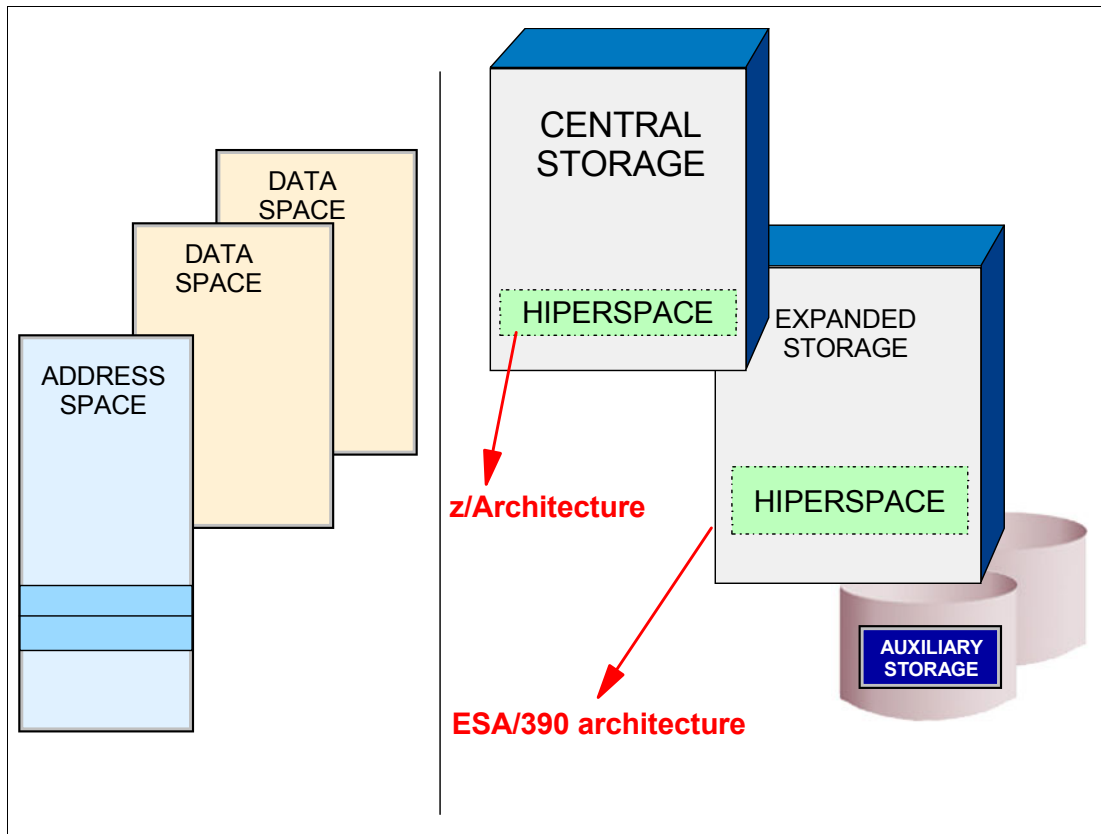


Figure 2-19 Data space management

The growth of processing, storage, and I/O capabilities led to a virtual storage constraint. The upward growth in virtual storage (address space size) was limited by the architecture (hardware and software).

ESA/370 and MVS/ESA came to relieve virtual storage constraint, bringing horizontal growth to virtual storage with *data space*, a new type of MVS address space, as well as *hiperspace*, a new type of service.

Data space is a type of space with a range up to 2 GB of contiguous virtual storage. The virtual storage map of a data space is quite different; except for the first 4K, the entire 2 GB is available for user data. A data space can hold only data; it does not contain MVS control blocks or programs in execution. Program code does not execute in a data space, although a program can reside in a data space as data. A program can refer to data in a data space at byte level, as it does in a work file.

A program references data in a data space directly, in much the same way it references data in an address space. It addresses the data by the byte, manipulating, comparing, and performing arithmetic operations. The program uses the same instructions (such as load, compare, add, and move character) that it would use to access data in its own address space. Before accessing data in a data space, a program must change its access mode, meaning, use special assembler instructions to change its *access mode*.

*High performance data access*—hiperspace—is a kind of data space created with the same RSM services used to create a data space. Hiperspace™ provides the applications an

opportunity to use expanded storage as a substitute to I/O operations. Hiperspaces differ from data spaces in the following ways:

- ▶ Main storage is never used to back the virtual pages in hiperspace.
- ▶ Data can be retrieved and stored between a hiperspace and a data space only using MVS services. This avoids the complex programming required when accessing data in a data space.
- ▶ Data is addressed and referred to as a 4K block.

Although z/OS and z/OS.e do not support Expanded Storage when running under the z/Architecture, hiperspace continues to operate in a compatible manner. Hiperspace under z/OS is in real storage.

Programs can use data spaces and hiperspaces to:

- ▶ Obtain more virtual storage than a single address space gives a user.
- ▶ Isolate data from other tasks in the address space. Data in an address space is accessible to all programs executing in that address space. You might want to move some data to a databases or hiperspace for security or integrity reasons. You can restrict access to data in those spaces to one or several units of work.
- ▶ Share data among programs that are executing in the same address space or different address spaces. Instead of keeping the shared data in common areas, create a databases or hiperspace for the data you want your programs to share. Use this space as a way to separate your data logically by its own particular use.
- ▶ Provide an area in which to map a data-in-virtual object.

## 2.20 64-bit address space map

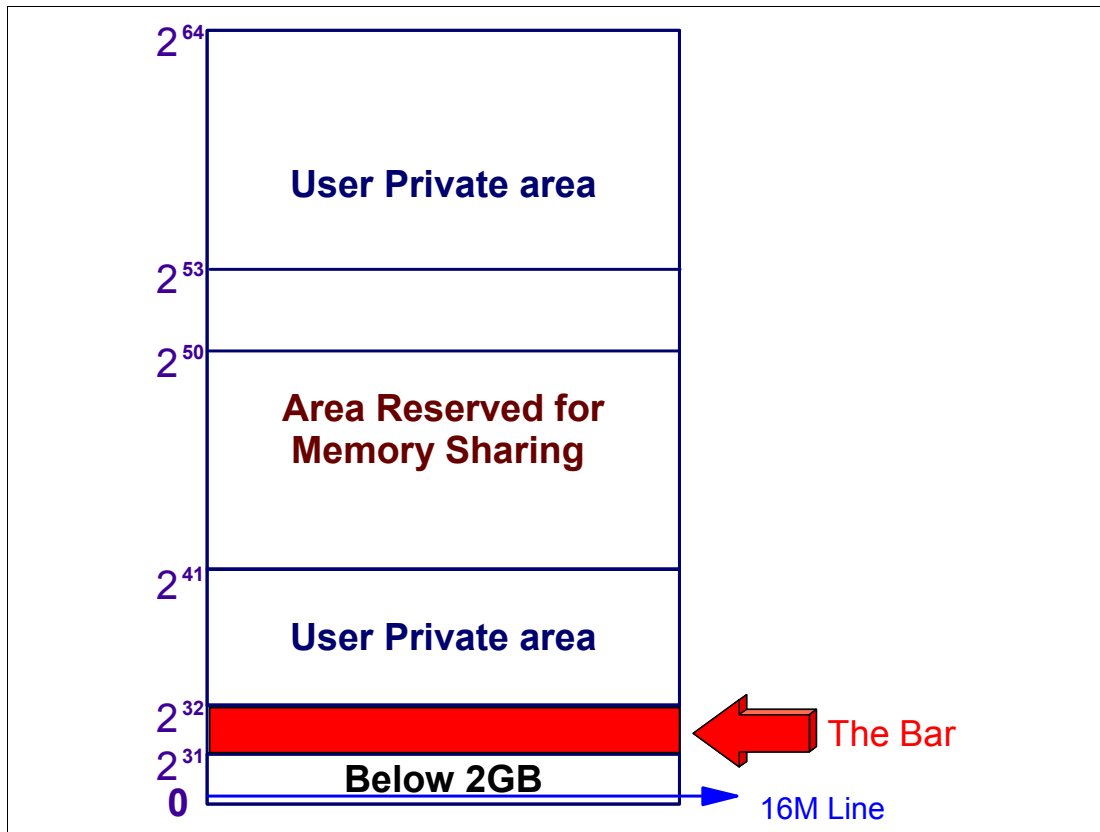


Figure 2-20 64-bits Address Space Map

The real storage 2 GB limit was broken with z/Architecture:

- ▶ OS/390 V2R10 and z/OS V1R1 support up to 128 GB of real storage when running in z/Architecture mode on an IBM 2064.
- ▶ z/OS V1R2 provides for up to 256 GB of central storage to be configured to a z/OS image.

The initial support for 64-bit *virtual* addressing was introduced in z/OS Version 1 Release 2. The size of the 64-bit address space is 16 exabytes, which makes the new address space 8 billion times the size of the former S/390 address space. Programs continue to be loaded and to run below the 2 gigabyte address; these programs can use data that resides above 2 gigabytes.

The address space is still created with a size of 2 GB. The address space only become bigger when a program allocates virtual storage above 2GB. To allocate and release virtual storage above 2G, a program must use the services provided in the IARV64 macro. The GETMAIN, FREEMAN, STORAGE, and CPOOL macros do not allocate storage above the 2 gigabyte address, nor do callable cell pool services.

For compatibility, the layout of the storage areas for an address space is the same below 2 GB, providing an environment that can support both 24-bit and 31-bit addressing. The area that separates the virtual storage area below the 2 GB address from the user private area is called the *bar*, as shown in Figure 2-20. In a 64-bit virtual storage environment, the terms “above the bar” and “below the bar” are used to identify the areas between  $2^{31}$  and  $2^{64}-1$ , and 0 and  $2^{31}-1$ , respectively. For example, any address in the range 0 to 7FFFFFFF is



below the bar, and an address in the range FFFFFFFF to 7FFFFFFF\_FFFFFFFF is above the bar. This is basically an alteration to the 2 GB 31-bit terminology that related “below the line” to 24-bit storage, and “above the line” to 31-bit addresses.

The 64-bit address space map is:

- ▶ **0 to 2\*\*31:** The layout is the same; see Figure 2-12 on page 58.
- ▶ **2\*\*31 to 2\*\*32:** From 2 GB to 4 GB is considered the *bar*. Below the bar can be addressed with a 31-bit address. Above the bar requires a 64-bit address. Just as the system does not back the page at 7FFFF000 in order to protect programs from addresses which can wrap to 0, the system does not back the virtual area between 2 GB and 4 GB. That means a 31-bit address with the high bit on will always program check if used in amode 64.
- ▶ **2\*\*31 - 2\*\*41:** The Low Non-shared area starts at 4G and goes to 2\*\*41.
- ▶ **2\*\*41 - 2\*\*50:** The Shared Area starts at 2\*\*41 and goes to 2\*\*50 or higher if requested.
- ▶ **2\*\*50 - 2\*\*64:** The High Non-shared area starts at 2\*\*50 or wherever the Shared Area ends and goes to 2\*\*64.

**Note:** The Shared Area will not be supported until the UNIX and C 64-bit support is added in a future release. It has been left in the picture at this time to show where the support is going.

## User private area

The area above the bar is intended for application data; no programs run above the bar. No system information or system control blocks exist above the bar, either. Currently there is no common area above the bar. However, IBM reserves an area above the bar to be used for future enhancements. The user can also reserve some area in the virtual storage allocated above the bar. This area is called the *guard area*. This area can be used for enhancements of user applications in the future.

The *user private area*, as shown in Figure 2-20 on page 68, includes:

- ▶ Low private: The private area below the line
- ▶ Extended private: The private area above the line
- ▶ Low Non-shared: The private area just above the bar
- ▶ High Non-shared: The private area above Shared Area

As users allocate private storage above the bar, it will first be allocated from the Low Non-shared area. Similarly, as Shared Area is allocated, it will be allocated from the bottom up. This is done to allow applications to have both private and shared memory above the bar and avoid additional machine cycles to perform dynamic address translation (DAT).

For virtual storage above the bar, a new JCL keyword, MEMLIMIT is introduced on the JOB and EXEC JCL statements. For virtual storage above the bar, there is no practical limit to the amount of virtual address range an address space can request. However, there are practical limits to the real storage and auxiliary storage needed to back the request. Therefore, a limit is placed on the amount of usable virtual storage above the bar an address space can use at any one time.

**Important:** MEMLIMIT controls the amount of usable storage above the 2 GB line.

## 2.21 Size and number notation

Symbol	Power of 2	Decimal Value
Kilo (K)	$2^{10}$	1024
Mega (M)	$2^{20}$	1,048,576
Giga (G)	$2^{30}$	1,073,741,824
Tera (T)	$2^{40}$	1,099,511,627,776
Peta (P)	$2^{50}$	1,125,899,906,842,624
Exa (E)	$2^{60}$	1,152,921,504,606,846,976

Figure 2-21 Size and number notation

With the introduction of 64-bit virtual addresses we are going into a new order of magnitude of numbers. We feel it is appropriate to refresh your mind on the names and raw sizes of numbers we now will start using.

The letters K, M, G, T, P, and E denote the multipliers  $2^{10}$ ,  $2^{20}$ ,  $2^{30}$ ,  $2^{40}$ ,  $2^{50}$ , and  $2^{60}$ , respectively. The letters are borrowed from the decimal system and stand for Kilo ( $10^3$ ), Mega ( $10^6$ ), Giga ( $10^9$ ), Tera ( $10^{12}$ ), Peta ( $10^{15}$ ), and Exa ( $10^{18}$ ). Note that Exa is 1 followed by 18 zeroes.

In the z/Architecture world these multiplier letters do not have the decimal meaning but instead represent the power of 2 closest to the corresponding power of 10. Figure 2-21 shows the names and the decimal values of these multipliers.

## 2.22 Segment tables and page tables

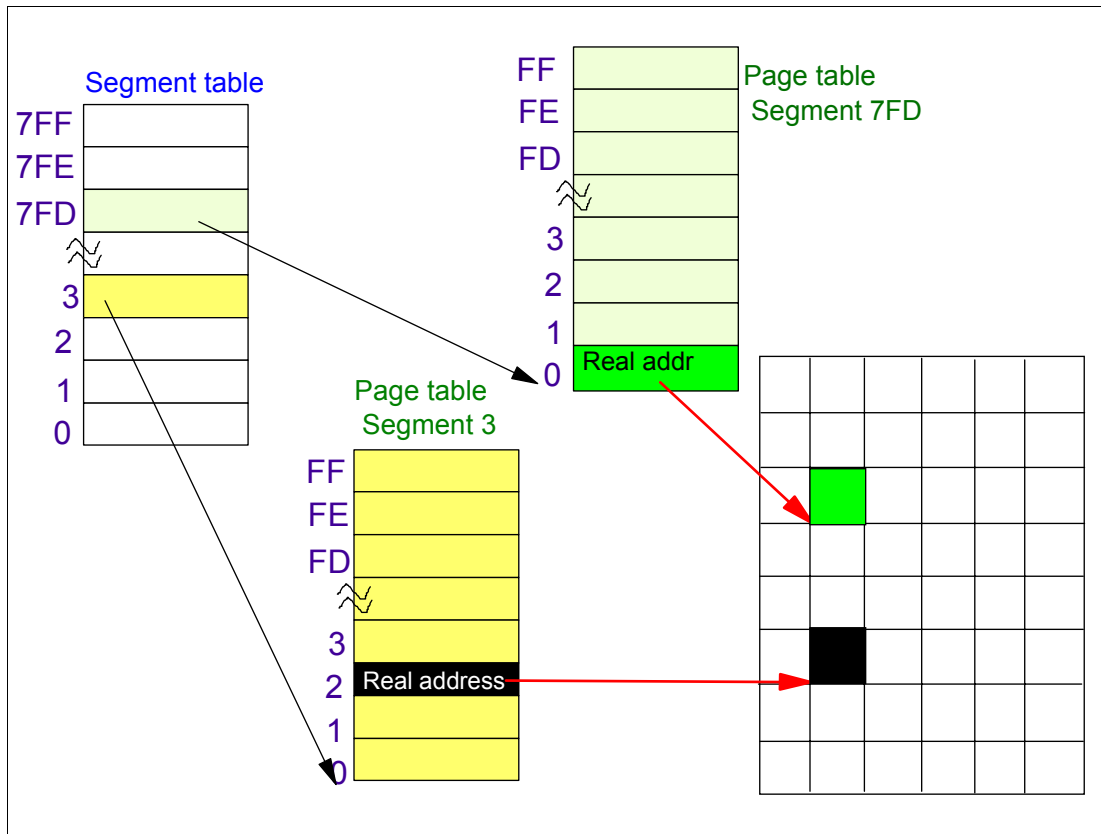


Figure 2-22 31-bit virtual address and dynamic address translation

Dynamic address translation is the process of translating a virtual address during a storage reference into the corresponding real address.

The virtual address space is partitioned into segments, each one of 1 MB. So, each 2 GB address space has 2048 segments. Now, to map each segment in a table, the system uses a table with 2048 entries. Each one is identified from 0 to 2047. The entry 0 refers to segment 0, the entry 1 refers to segment 1, and so forth. To identify each segment requires 11 bits:  $2047 = b'111\ 1111\ 1111' = X'7FF'$ . This table is called a *Segment table*, and each address space has one.

Since each page is 4K, each address space segment has 256 pages. To map each page of a segment in into a table, the system uses a table with 256 entries. Each entry identifies each page in that segment. Each entry is identified from 0 to 255. So, entry 0 refers to the first page of the segment, entry 1 refers to the second page in the same segment, and so forth. To identify each page requires 8 bits:  $255 = b'1111\ 1111' = X'FF'$ . This table is called a *Page table*.

Each address space has its own Segment table and Page tables. Each Segment table entry has a pointer to the correlate Page table. There are pointers only for those Page tables having pages with getmained addresses. A Page Table is allocated when the first page on that segment is allocated. There are a maximum of 2048 Page tables.

The DAT (Dynamic Address Translation) is a hardware component responsible for translating a virtual address into a real address. The MVS maintains the Segment and Page Tables.

## 2.23 31-bit virtual address

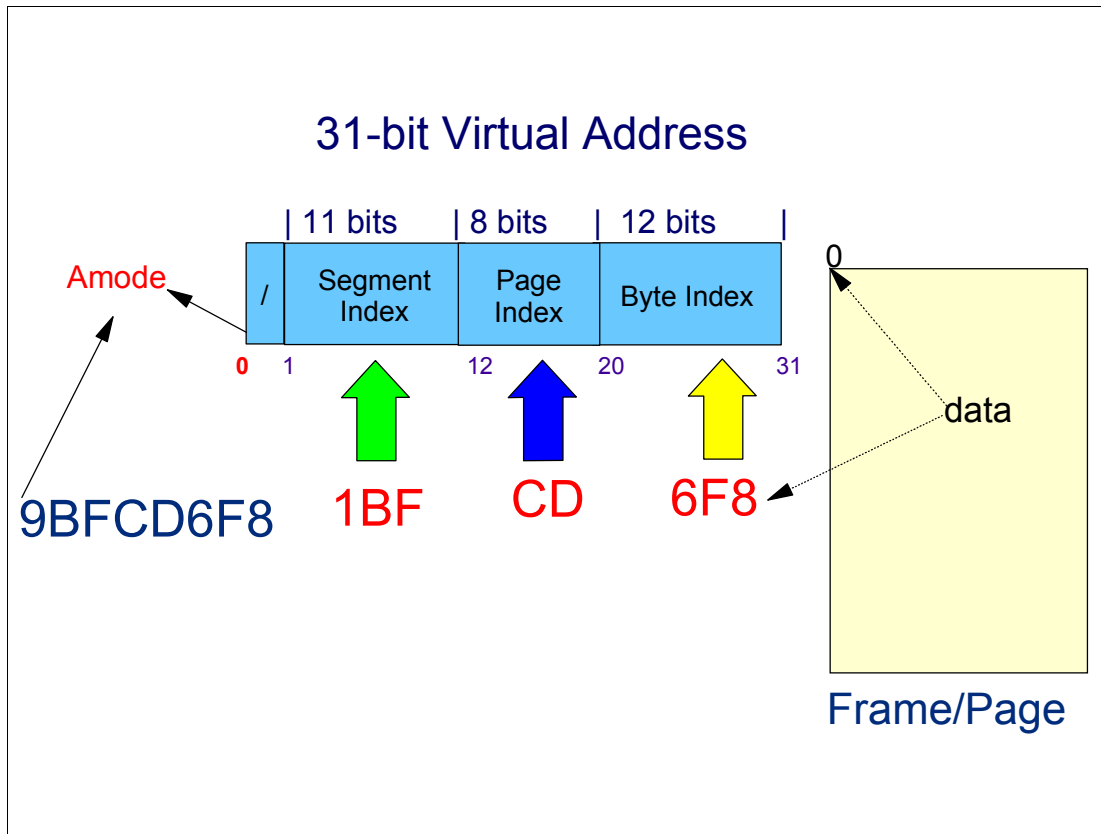


Figure 2-23 Translating 31-bit virtual addresses

Putting all together, lets read a virtual address:

- ▶ Bit 0: not used
- ▶ Bits 1 - 11 identify the segment number
- ▶ Bits 12 - 19 identify the page number in that segment
- ▶ Bits 20 - 31 indicate the displacement of the data in that page

To translate a 31-bit virtual address (2G) into a real address, DAT uses:

1. Bits 1 - 11 as an index in the Segment table to find the Page table address of that segment
2. Bits 12 - 19 as an index in the Page table to the find the frame address
3. Bits 20 - 31 as the displacement of the data from the beginning of the frame

## 2.24 64-bit virtual address translation

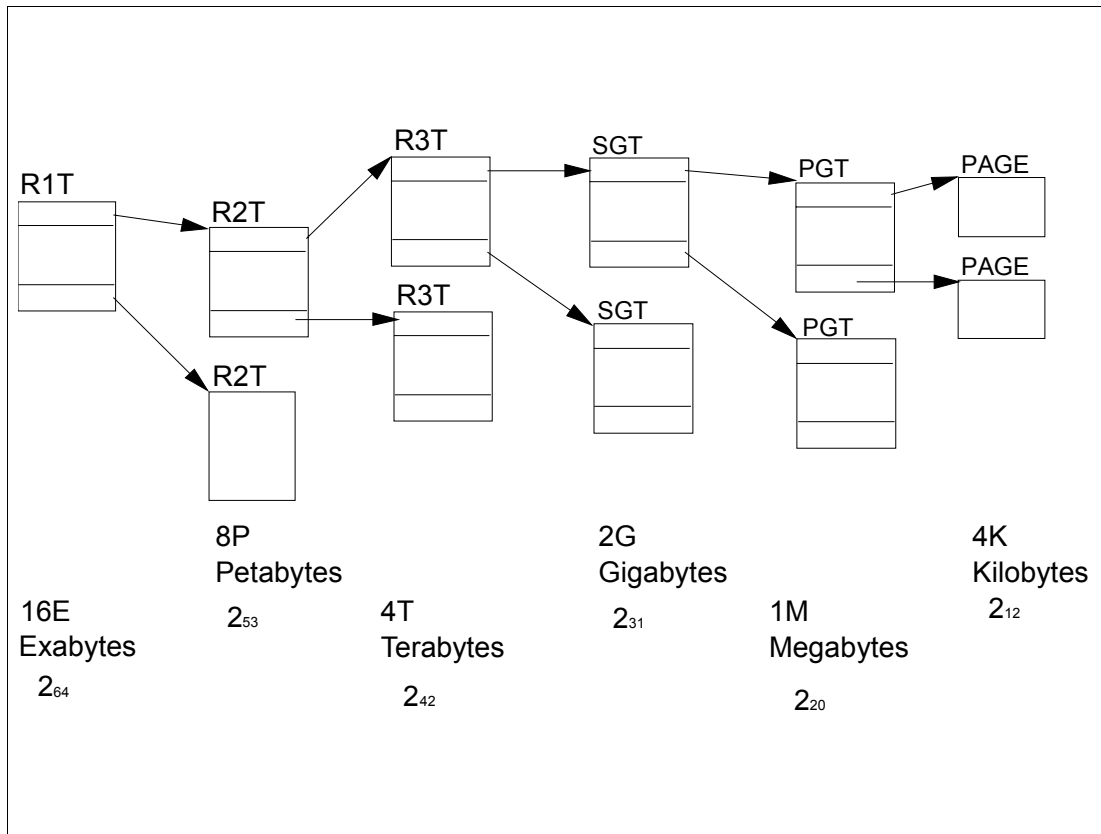


Figure 2-24 64-bit dynamic address translation

In a 16 EB address space with 64-bit virtual storage addressing, there are three additional levels of translation tables, called *Region tables*. They are called region third table (R3T), region second table (R2T), and region first table (R1T). The Region tables are 16 KB in length, and there are 2048 entries per table. Each region has 2G bytes.

When the first storage is created above the bar, RSM creates the R3T. The R3T table has 2048 segment table pointers, and provides addressability to 4 TB. When virtual storage greater than 4 TB is allocated, an R2T is created. An R2T has 2048 R3T table pointers and provides addressability to 8 PB. An R1T is created when virtual storage greater than 8 PB is allocated. The R1T has 2048 R2T table pointers and provides addressability to 16 EB. Figure 2-24 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, once you have identified the corresponding 2G region entry that points to the Segment table, the process is the same as that described previously.

RSM only creates the additional levels of region tables when necessary to back storage which is mapped. They are not built until a translation exception occurs. 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 may 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.

## 2.25 Translating a 64-bit virtual address

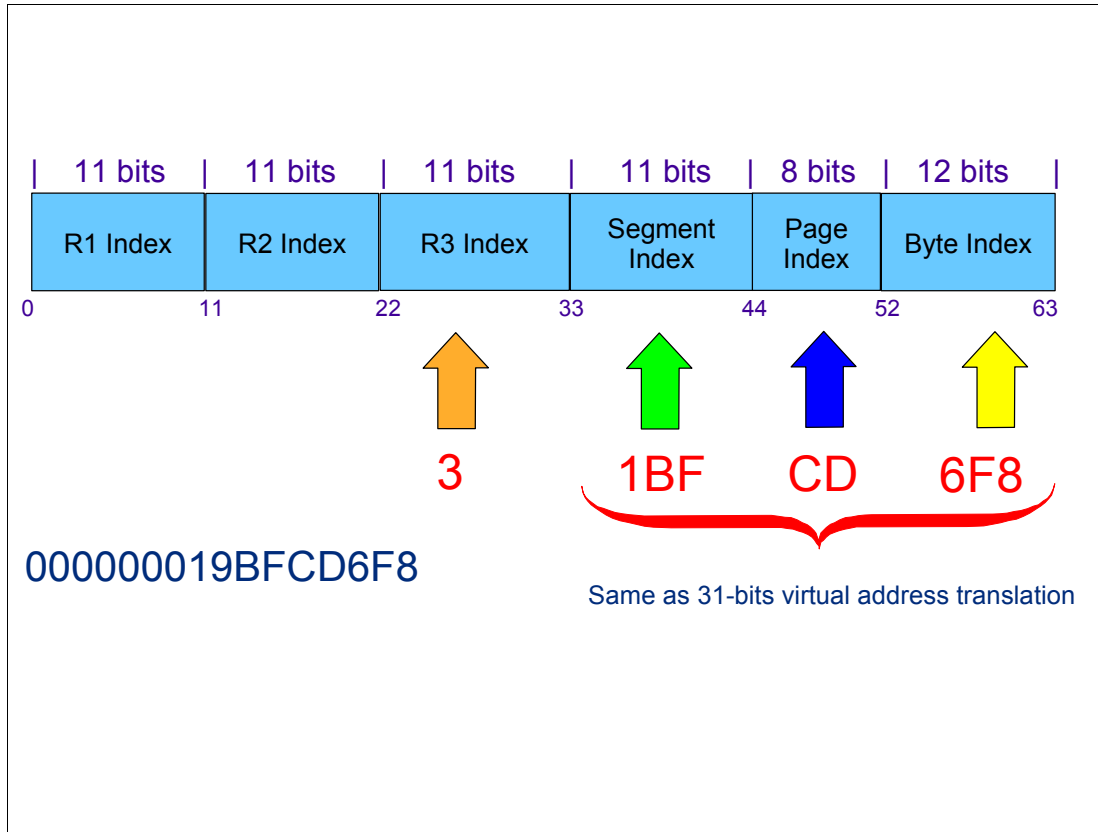


Figure 2-25 Translating a virtual address

To translate a 64-bit virtual address into a real address, DAT uses:

- ▶ Bits 0 - 10 are the first region index into the R1T table.
- ▶ Bits 11 - 21 are the second region index into the R2T table.
- ▶ Bits 22 - 32 are the third region index into the R3T table.
- ▶ Bits 33 - 43 are the segment index into the segment table.
- ▶ Bits 44 - 51 are the page index into the page table
- ▶ Bits 52 - 63 indicate the data displacement into the page itself.

With 64-bit virtual addressing, we now get three more 11-bit region indexes into the three region tables, as shown in Figure 2-25.

RSM only creates the additional levels of region tables when necessary to back storage which is mapped.

## 2.26 System initialization

- Starting MVS
  - Initialization process
    - Starting system address spaces
    - Starting other address spaces

Figure 2-26 System initialization

To tailor the system's storage parameters, you need a general understanding of the system initialization and storage initialization processes.

The system initialization process prepares the system control program and its environment to do work for the installation. The process essentially consists of:

- ▶ System and storage initialization, including the creation of system component address spaces.
- ▶ Master scheduler initialization and subsystem initialization.

When the system is initialized and the job entry subsystem is active, the installation can submit jobs for processing by using the START, LOGON, or MOUNT command.

In addition to initializing system areas, MVS establishes system component address spaces. MVS establishes an address space for the master scheduler (the \*MASTER\* address space) and other system address spaces for various subsystems and system components. Some of the system component address spaces are:

- ▶ Program call/authorization for cross-memory communications
- ▶ System trace
- ▶ Global resource serialization
- ▶ Dumping services

## 2.27 z/OS address spaces

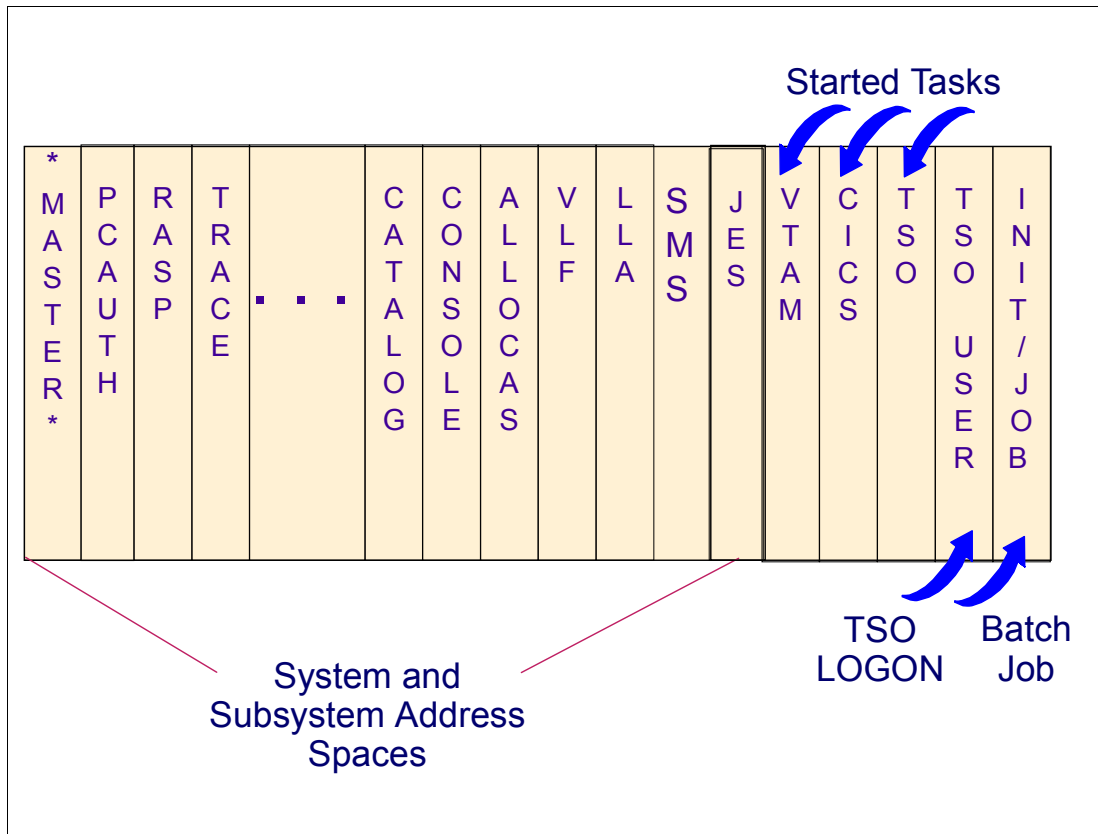


Figure 2-27 System and subsystem address spaces in z/OS

When you start z/OS, master scheduler initialization routines initialize system services such as the system log and communications task, and start the master scheduler address space (\*MASTER\*). Each address space created has a number associated to it, called the address space id (ASID). Since the master scheduler is the first address space created in the system, it becomes address space number one (ASID=1). Other system address spaces are then started during the initialization process of z/OS.

Then the subsystem address spaces are started. The master scheduler starts the job entry subsystem (JES2 or JES3). JES is the primary job entry subsystem. Then other defined subsystems are started. All subsystems are defined in SYS1.PARMLIB, member IEFSSNxx. These subsystems are secondary subsystems.

Figure 2-27 shows four types of address spaces as follows:

- System** The system address spaces are started following initialization of the master scheduler. These address spaces perform functions for all the other types of address spaces that start in a z/OS system.
- Subsystem** You cannot run z/OS without a primary job entry subsystem, either JES2 or JES3. SMS is also a subsystem.
- TSO logon** These address spaces start when a user issues a logon to TSO/E. Each TSO user executes in a separate address space.
- Batch job** These address spaces are started by JES when a JCL stream is passed to JES and a job is created and then subsequently scheduled into execution.



## 2.28 Subsystem definitions

### SYS1.PARMLIB: IEFSSNxx

```
SUBSYS      SUBNAME (subname)
             [CONSNAME (consname) ]
             [INITRTN (initrtn)
             [INITPARM (initparm) ] ]
             [PRIMARY ({NO | YES})
             [START ({YES | NO}) ] ]
```

```
SUBSYS SUBNAME(JES2) PRIMARY(YES)
```

```
SUBSYS SUBNAME(SMS) INITRTN(IGDSSIIN)
        INITPARM('ID=60,PROMPT=YES')
```

### SYS1.PARMLIB: IEASYSxx SSN=xx

Figure 2-28 Subsystem definitions in SYS1.PARMLIB

Subsystem initialization is the process of readying a subsystem for use in the system. IEFSSNxx members of SYS1.PARMLIB contain the definitions for the primary subsystems, such as JES2 or JES3, and the secondary subsystems, such as SMS, CICS, DB2, and so forth. For detailed information about the data contained in IEFSSNxx members for secondary systems, refer to the installation manual for the specific subsystem. IEFSSNxx allows you to specify the following:

- ▶ The subsystem initialization routine to be given control during master scheduler initialization.
- ▶ The input parameter string to be passed to the subsystem initialization routine.
- ▶ A primary subsystem name and whether you want it started automatically.

The order in which the subsystems are initialized depends on the order in which they are defined in the IEFSSNxx parmlib member on the SSN parameter. Unless you are starting the storage management subsystem (SMS), start the primary subsystem (JES) first.

**Note:** The storage management subsystem (SMS) is the only subsystem that can be defined before the primary subsystem.

Some subsystems require the services of the primary subsystem in their initialization routines. Problems can occur if subsystems that use the subsystem affinity service in their initialization routines are initialized before the primary subsystem. If you are starting SMS, specify its record before you specify the primary subsystem record.

**Note:** In general, it is a good idea to make the subsystem name the same as the name of the member of SYS1.PROCLIB used to start the subsystem. If the name does not match, you may receive error messages when you start the subsystem.

The SSN parameter in IEASYSxx identifies the IEFSSNxx member that the system is to use to initialize the subsystems, as follows:

```
SSN=          {aa          }  
              {(aa,bb,...) }
```

The two-character identifier, represented by aa (or bb, and so forth), is appended to IEFSSN to identify IEFSSNxx members of parmlib. If the SSN parameter is not specified, the system uses the IEFSSN00 parmlib member.

The order in which the subsystems are defined on the SSN parameter is the order in which they are initialized. For example, a specification of SSN=(13,Z5) would cause those subsystems defined in the IEFSSN13 parmlib member to be initialized first, followed by those subsystems defined in the IEFSSNZ5 parmlib member.

**Note:** If you specify duplicate subsystem names in IEFSSNxx parmlib members, the system issues message IEFJ003I to the SYSLOG, the master console, and consoles that monitor routing code 10 messages.

## 2.29 Multiprogramming and multiprocessing

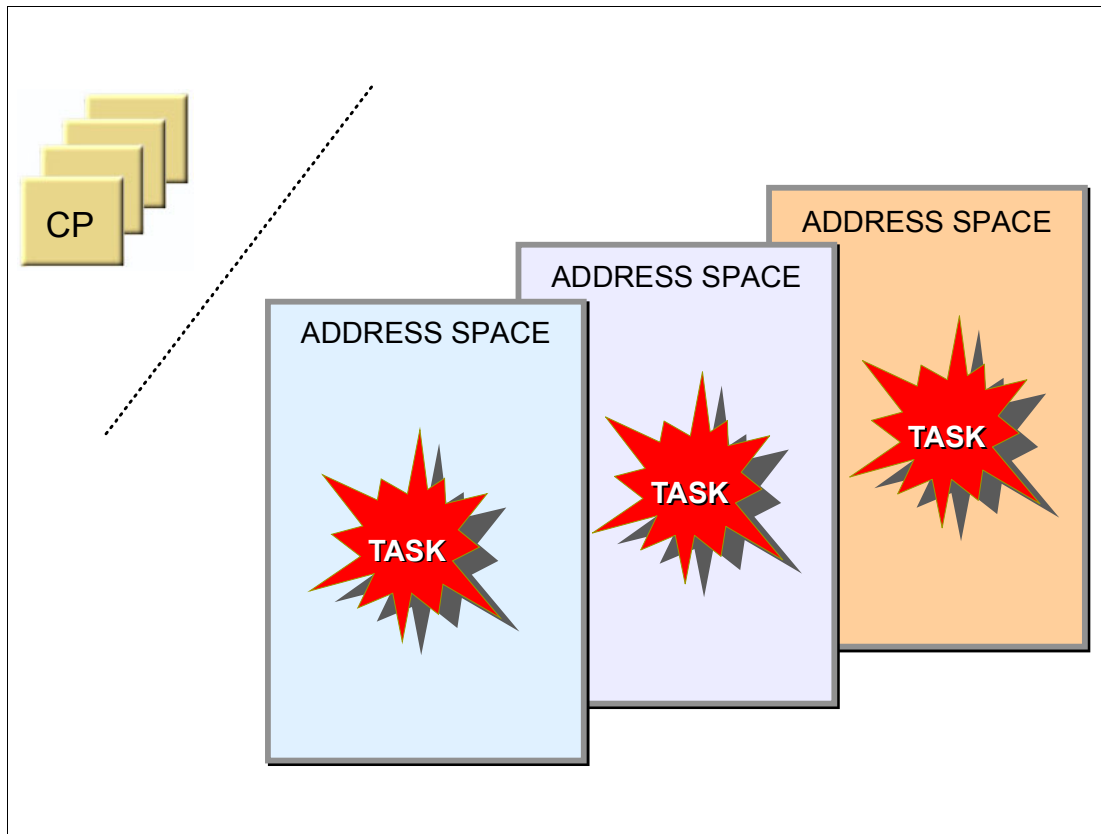


Figure 2-29 Multiprogramming and multiprocessing

### Multiprogramming

Many programs may be in the system at the same time, each in its own address space. In a single processor only one of these programs can be active at a time. However, the active program may lose control at any time because of interrupts. The SCP selects which program should get control next.

### Multiprocessing

Multiprocessing is a logical expansion of multiprogramming. It means the execution of more than one program (task) simultaneously on more than one processor. All processors operate under a single SCP.

Remember:

- ▶ Each processor has a current Program Status Word (PSW), its own set of registers, and assigned storage locations.
- ▶ When a single processor shares real storage with other processors, then all of them are controlled by a single SCP. This is called a *tightly coupled multiprocessing complex*.
- ▶ When a single processor shares a common workload with others, but does not share real storage, this is called a *loosely coupled multiprocessing complex*.

## 2.30 Program compile, link-edit, and execution

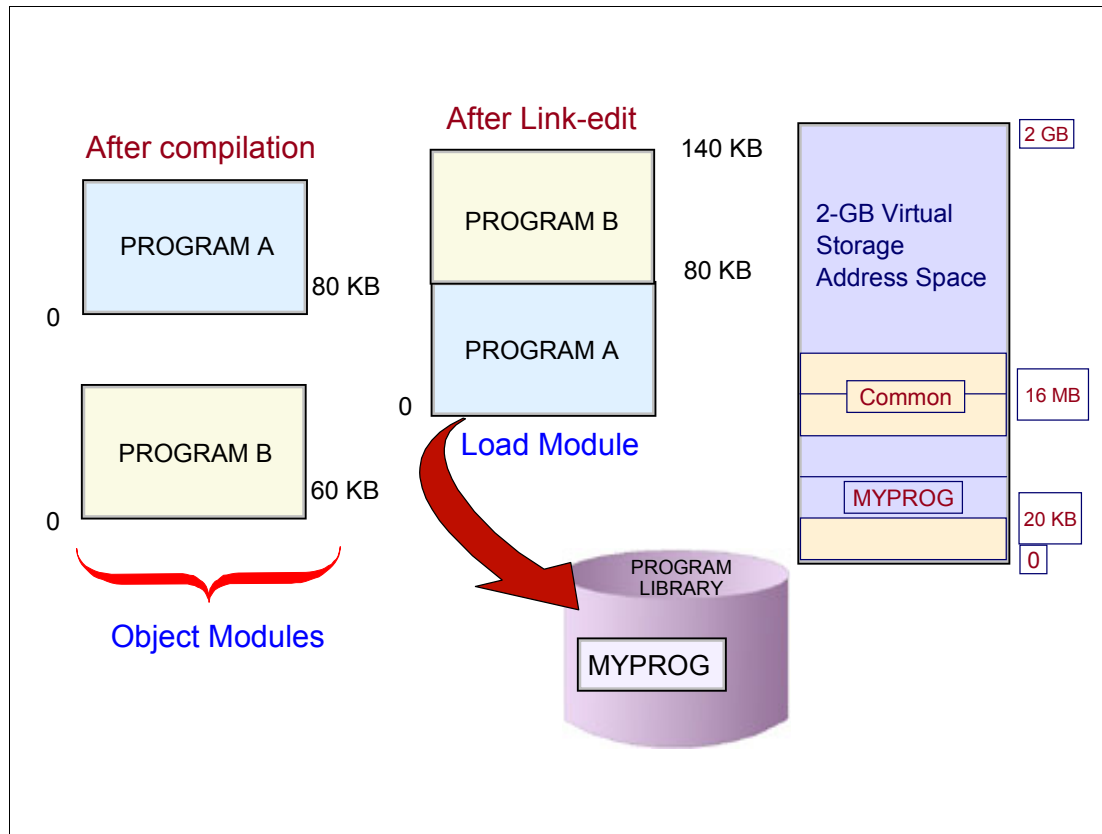


Figure 2-30 Executing a program after compiling and link-editing

A z/OS system may appear to be one big block of code that drives your CPU. Actually, z/OS is a complex system comprised of many different smaller blocks of code. Each of those smaller blocks of code perform a specific function within the system.

Each module of symbolic language code is first assembled or compiled by one of the language translators. The input to a language translator is a source module. The output from a language translator is an object module.

The linkage editor is a processing program that accepts object modules, load modules, control statements, and options as input. It combines these modules, according to the requirements defined by the control statements and options, into a single output load module that can be stored in a partitioned data set program library and loaded into storage for execution by the program management loader.

Each system function is composed of one or more load modules. In a z/OS environment, a load module represents the basic unit of machine-readable executable code. Load modules are created by combining one or more object modules and processing them with a link-edit utility. The link-editing of modules is a process that resolves external references and addresses. The functions on your system, therefore, are one or more object modules that have been combined and link-edited.



## **TSO/E, ISPF, JCL, and SDSF**

This chapter describes how to use the basic products a system programmer needs to install and customize a z/OS operating system, as follows:

- ▶ Time Sharing Option/Extensions (TSO/E)
- ▶ Interactive System Productivity Facility and Program Development Facility (ISPF/PDF)
- ▶ Job Control Language (JCL)
- ▶ Spool Display and Search Facility (SDSF)

## 3.1 z/OS facilities for system programmers

- ❑ Time Sharing Option/Extended - (TSO/E)
- ❑ Interactive System Productivity Facility - (ISPF)
- ❑ Job Control Language - (JCL)
- ❑ Spool Display and Search Facility - (SDSF)

*Figure 3-1 z/OS facilities for system programmers*

As a system programmer, you should be familiar with the basic tools used in your daily job. These tools and their uses are as follows:

- ▶ TSO/E and ISPF are used to:
  - Install and customize z/OS and other products
  - Communicate interactively with the operating system
  - Define and maintain user definitions
  - Create data sets and JCL, and submit jobs
  - Communicate with other TSO/E users
  - Develop and maintain programs in languages such as assembler, COBOL, FORTRAN, Pascal, C, C++, JAVA, PL/I, REXX, CLIST, and so on.
  - Manipulate data
- ▶ JCL enables you to submit jobs and allocate resources.
- ▶ SDSF is used to:
  - Monitor
    - JOBS waiting for execution
    - Output waiting to be printed
    - System resources used by JOBS
    - System resources available for utilization
  - Control
    - System resources like printers
    - JOB priority and class
    - JOB output priority and class

## 3.2 TSO/E

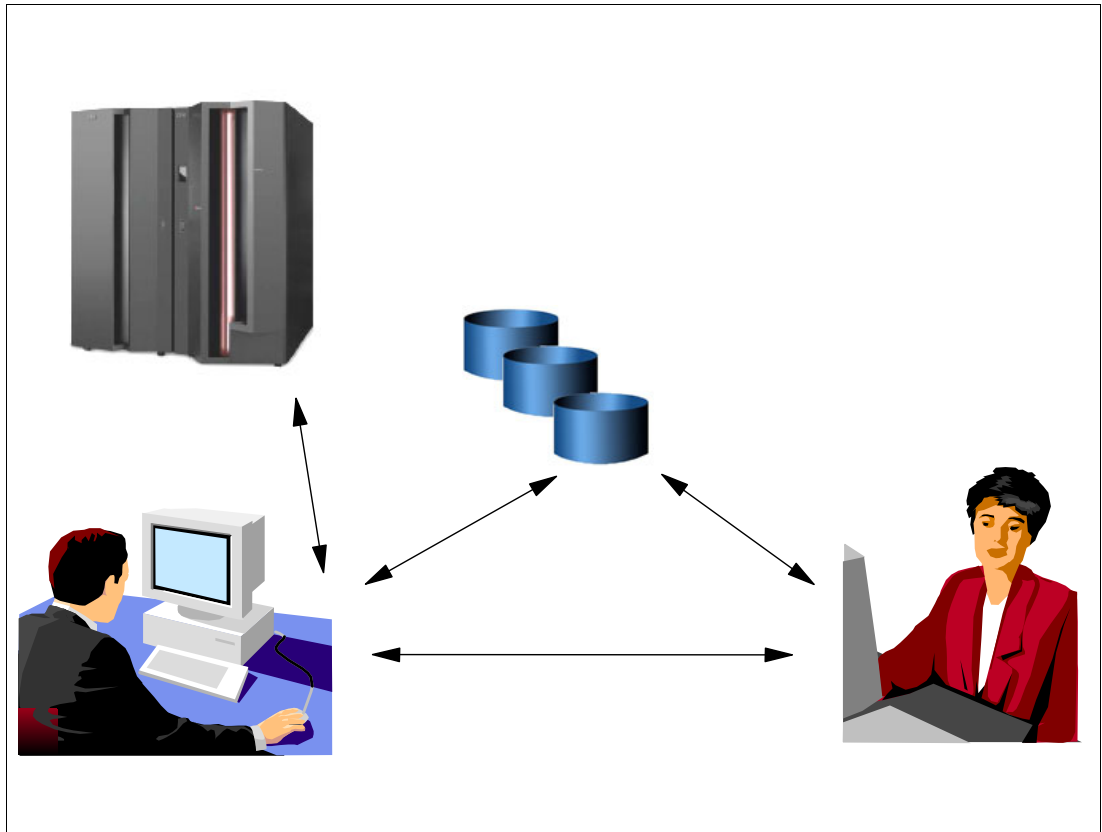


Figure 3-2 TSO/E

TSO/E is a base element of z/OS, as it was of OS/390 and of the former MVS systems. TSO/E has undergone continuous enhancements during its life, and it has become the primary user interface to the OS/390 system and, now, to the z/OS system.

TSO/E provides programming services that you can use in system or application programs. These services consist of programs, macros, and CLISTs. TSO/E services support a wide range of functions that are useful in writing system programs as well as application programs that exploit the full-screen capabilities of TSO/E.

CLISTs, REXX execs, servers, and command processors are specific types of programs that you can write to run in the TSO/E environment.

## 3.3 TSO/E highlights

- Session Manager
- Commands
- Online help
- Console
- Support for z/OS Unix
- CLIST
- REXX
- Data and notice handling
- Security

Figure 3-3 Key features of z/OS TSO/E

Some highlights of TSO/E are:

► The Session Manager

The TSO/E Session Manager is an interface to line mode TSO/E. It saves the commands that you enter and the responses that you receive and allows you to re-display or print them. You can correct or change a command that is displayed on the screen without having to retype the entire command. By allowing you to re-display, change, and reuse your input, the Session Manager makes TSO/E easier to use.

► Commands

TSO/E provides numerous commands for both end users and programmers that allow them to interact with TSO/E and the MVS system. The **ALLOCATE**, **FREE**, and **EDIT** commands are examples of commands that allow users to manage their data sets. The **TEST** and **TESTAUTH** commands let programmers test assembler language programs, including command processors, APPC/MVS transaction programs, and other programs written in assembler language. The **CONSOLE** command lets users with **CONSOLE** command authority perform MVS operator activities from a TSO/E session.

► Online help

Terminal users can obtain online help for most TSO/E commands. Information Center Facility users can obtain help for each panel and message. TSO/E Enhanced Connectivity Facility users can also obtain online help for terminal messages. Installations can also provide online help information to users in different languages.



► Data and notice handling

TSO/E simplifies the way in which data and notices are sent and received. For example, the **TRANSMIT** and **RECEIVE** commands let users send data and messages to other users in a network. The broadcast data set or individual user logs contain messages that either the system or another user sends using the **SEND** command. In addition, a recovery routine prevents broken mail chains that could occur when message handling is interrupted. Notices are also handled more efficiently during logon processing. TSO/E keeps a copy of notices in storage, thereby reducing the I/O operations needed to inform users of waiting messages when they log on.

► Logon processing

TSO/E provides a full-screen logon panel that makes the logon process easier by:

- Saving user attributes from one session to the next
- Allowing program function keys to be used during logon
- Allowing users to enter commands during logon
- Explaining the error when incorrect information is specified

The **LOGON** and **ACCOUNT** command processors allow users to request private areas of up to 2 gigabytes for each terminal session. Your installation can also customize the logon panel and the logon help panel, and customize logon processing using different exits.

► Language enablement

TSO/E allow installations to provide TSO/E messages and the **TRANSMIT** full-screen panel to users in different languages. The TSO/E **CONSOLE** command also supports the display of translated system messages issued during a console session.

The logon authorized pre-prompt exit and the **PROFILE** command support the specification of languages to be used in displaying translated information. An installation can specify help data sets for different languages in the IKJTSOxx member of SYS1.PARMLIB. Support for logon panels and their help text in different languages is also available.

The TSO/E REXX external function, **SYSVAR** provides support for new arguments that REXX execs can use to obtain language information. Execs can use this information together with the new **SETLANG** function to set the language in which REXX messages are displayed.

► Security

TSO/E provides several enhancements to support the use of security labels. Installations can also control communication between users to protect the security classification of information. For example, installations can control and audit the use of the **SEND** command. **LISTBC** command processing enhancements let installations restrict users from viewing messages for which they do not have the proper security.

► CLIST language

The CLIST language is a high-level programming language that lets programmers issue lists of TSO/E commands and JCL statements in combination with logical, arithmetic, and string-handling functions provided by the language. The programs, called CLISTs, can simplify routine user tasks, invoke programs written in other languages, and perform complex programming functions by themselves.

► Restructured Extended Executor (REXX) language support

REXX is a high-level procedures language that enables inexperienced users as well as experienced programmers to write structured programs called REXX execs. REXX execs can be executed in any MVS address space (both TSO/E and non-TSO/E). TSO/E also allows users to write APPC/MVS transaction programs in the REXX language.

Installations can acquire IBM Compiler and Library for REXX/370 (licensed program number 5695-013) or a functionally equivalent compiler. A compiled REXX exec executes more efficiently because the exec does not need to be interpreted at run time.

► The TSO/E service facility

The TSO/E service facility lets TSO/E users execute authorized or unauthorized programs, TSO/E commands, or CLISTs from an unauthorized environment, while maintaining system integrity.

► TSO Command Package

The TSO Command Package provides functions that help to improve productivity. The functions included are:

- Support for running terminal sessions as batch jobs
- Automatic saving of data
- Accounting facility
- Defaults for the user-attribute data set

► The Information Center Facility

The Information Center Facility eases users into the data processing environment by providing a series of conversational panels. These panels eliminate numerous command-driven interactions between the user and the system. Information Center Facility provides panels that enable an administrator to maintain the facility, enroll users, and add, modify, and delete products.

► The Enhanced Connectivity Facility

The Enhanced Connectivity Facility (ECF) allows you to customize the way in which host server programs and PC requester programs communicate. IBM products or customer-written programs can supply the services.

The user can access MVS host services from a PC using IBM System/370-to-IBM Personal Computer ECF. This allows a DOS/PC user to interact with MVS or VM/SP systems using PC commands.

► Support for z/OS UNIX

Installations can use the functions provided by the TSO/E **ALLOCATE** and **FREE** commands to manipulate z/OS UNIX files.

## 3.4 TSO/E customization

- Define TSO/E to VTAM or TCAM
  
- Define the users allowed to log on to TSO/E
  
- Create TSO/E logon procedure for users

Figure 3-4 TSO/E required customization

TSO/E allows users to interactively work with the MVS system. After the required customization, users are able to log on and issue commands from TSO/E.

Each user is defined to TSO/E by storing its user ID, logon procedure name, and the TSO/E resources which it has authority to use. This can be done in either of two ways:

- ▶ User Attribute Data Set (UADS), using the **ACCOUNT** command
- ▶ RACF database

When RACF is installed, it can be used to control access to the system and store information about each TSO/E user. The RACF database contains profiles for every entity (user, data set, or group) defined to RACF. For more information about RACF, see *z/OS V1R4.0 Security Server RACF System Programmer's Guide*, SA22-7681.

Customization of the TSO/E environment generally refers to making a TSO/E facility available or changing default values that affect TSO/E. You can customize:

- ▶ VTAM: You can change VTAM session protocols, provide substitute characters for unavailable keyboard characters, and override the default values used to start VTAM.
- ▶ TCAM: You can override the default values used to start TCAM.
- ▶ Logon limits: You can limit and manage the maximum number of concurrent logons, the user's region size, and user access to applications.

**Note:** In z/OS.e, the number of concurrent TSO/E sessions is limited to eight.

- ▶ The logon/logoff process: You can change how often the system displays the logon proceeding message, limit the number of attempts a user can make at entering information in response to logon prompts, tailor the reconnect option, and suppress messages that are generated during the execution of the logon job. You can also review factors that affect logon performance, such as using STEPLIBs in logon JCL, and you can write exits to further customize the logon/logoff process. Your installation can use security labels (SECLABEL) if the proper products are installed.
- ▶ ISPF/PDF and others products: You can make ISPF/PDF and others products that run in the TSO environment available to TSO/E users.
- ▶ Authorized commands and programs: You can select which authorized commands and programs users can use.
- ▶ Command/program invocation platform support: You can invoke TSO/E commands and programs on the command/program invocation platform. Both authorized and unauthorized commands and programs are supported.
- ▶ Command availability in the background: You can make specific commands unavailable for use in the background.
- ▶ TRANSMIT and RECEIVE availability: You can make the TRANSMIT and RECEIVE commands available.
- ▶ HELP data set usage: You can customize the use of HELP data set members.
- ▶ Host services availability: You can make host services available to personal computer (PC) users.
- ▶ Language support: You can provide information to users in their national language.

For more details about customization, refer to *z/OS TSO/E Customization*, SA22-7783.

## 3.5 TSO/E: TCAS start procedure

```
//TSO      PROC MBR=TSOKEY00
/** LIB: CPAC.PROCLIB(TSO)
/** DOC: THIS IS THE CATALOGED PROCEDURE USED
/**      FOR STARTING TSO/VTAM TIME SHARING.  THE
/**      TERMINAL CONTROL ADDRESS SPACE (TCAS) MUST
/**      BE ACTIVE WHEN RUNNING TSO/VTAM TIME SHARING.
/**
/** NOTE: THE PARMLIB DD STATEMENT IDENTIFIES THE PARMLIB
/**      DATA SET AND MEMBER THAT CONTAINS TSO/VTAM TIME
/**      SHARING PARAMETERS.  THE MEMBER MAY EITHER BE
/**      SPECIFIED ON THE START COMMAND OR DEFAULTED TO
/**      TSOKEY00.
/**
/**      THE PRINTOUT DD STATEMENT IDENTIFIES WHERE THE
/**      TIME SHARING PARAMETERS THAT ARE USED SHOULD BE
/**      LISTED.
/**
/**      FREE=CLOSE IS CODED SO THAT THE DATA SETS ASSOCIATED
/**      WITH BOTH THE PARMLIB AND PRINTOUT DD STATEMENTS
/**      WILL BE DEALLOCATED WHEN THE DATA SETS ARE CLOSED.
/**
//STEP1   EXEC PGM=IKTCAS00,TIME=1440
//PARMLIB DD DSN=CPAC.PARMLIB(&MBR),DISP=SHR,FREE=CLOSE
//PRINTOUT DD SYSOUT=*,FREE=CLOSE
/**
```

Figure 3-5 TSO/E: TCAS start procedure

Before a user can log on to TSO/E, both VTAM and the terminal control address space (TCAS) must be active in the system. The system operator enters the **START** command to start VTAM.

Once VTAM has been started, the system operator enters the **START** command to start TSO/E and activate TCAS. TCAS accepts logons from TSO/VTAM users and creates an address space for each user.

The TCAS (TSO) start procedure is usually stored at SYS1.PROCLIB. In the start procedure you specify the TSOKEYxx SYS1.PARMLIB member that contains the parameters to be used by TCAS to control the time-sharing buffers, maximum number of users, and other operational variables, as pointed by the PARMLIB DD card of the start procedure. If the PARMLIB DD card is not coded in the procedure, SYS1.PARMLIB is used.

When a user logs on, the VTAM terminal I/O coordinator (VTIOC) is initialized. VTIOC controls the movement of data between TSO/E and VTAM. The Parmlib member TSOKEY00 or an installation-defined alternate member contains parameters that are used during VTIOC initialization. If a member other than TSOKEY00 is used, the operator must include the member name either on the **START** command or in the procedure that the **START** command invokes. For a description of TSOKEY00, see *z/OS V1R4.0 MVS Initialization and Tuning Reference*, SA22-7592.

## 3.6 TSO/E logon procedure

```
/*-----  
/* SERVERPAC LOGON PROCEDURE  
/*  
/* THIS PROCEDURE ENABLES USERS TO LOG ON TO TSO/E.  
/* THE CLIST ISPPDF, WHICH RESIDES IN CPAC.CMDPROC,  
/* IS EXECUTED AT FIRST TO INVOKE THE ISPF.  
/*-----  
//IKJACCNT PROC  
//IKJACCNT EXEC PGM=IKJEFT01,DYNAMNBR=500,PARM=ISPPDF  
//SYSPROC DD DISP=SHR,DSN=CPAC.CMDPROC  
//SYSHELP DD DISP=SHR,DSN=SYS1.HELP  
// DD DISP=SHR,DSN=SYS1.SASPHLP  
// DD DISP=SHR,DSN=ISF.SISFHELP  
// DD DISP=SHR,DSN=REXX.V1R3M0.SEAGHENU  
// DD DISP=SHR,DSN=REXX.V1R3M0.SFANHENU  
// DD DISP=SHR,DSN=SYS1.SBDTHELP  
// DD DISP=SHR,DSN=SYS1.HELPEXP  
// DD DISP=SHR,DSN=ISP.SISPHELP  
//SYSLBC DD DISP=SHR,DSN=SYS1.BROADCAST  
//SYSPRINT DD TERM=TS,SYSOUT=*  
//SYSTEM DD TERM=TS,SYSOUT=*  
//SYSIN DD TERM=TS  
/*
```

Figure 3-6 TSO/E logon procedure

A TSO/E logon procedure contains JCL statements that execute the required program and allocate the required data sets to enable a user to acquire the resources needed to use TSO/E. To log on to TSO/E, a user must have access to at least one logon procedure.

The logon procedure is usually located in data set SYS1.PROCLIB or another library identified in the PROCxx concatenation in the JES2 startup procedure, or in the IATPLBxx DD statement in the JES3 startup procedure. TSO/E provides a logon procedure in SYS1.PROCLIB called IKJACCNT for system programmers to access the system, for example, during the initial installation or if there are problems with the RACF database. Figure 3-6 shows a sample logon procedure. The statements specify:

- PGM=IKJEFT01** Identifies the program to be executed. IKJEFT01 is the TSO/E-supplied Terminal Monitor Program (TMP) that provides an interface between the user command processors and the TSO/E control program. It obtains commands, gives control to command processors, and monitors their execution. This program can also be executed in the background by submitting JCL. Instead of the IKJEFT01 program, an installation can use the Session Manager program (ADFMDF03) or its own terminal monitor program.
- PARM** You can pass to IKJEFT01 a command, CLIST, REXX, or a program to be interpreted as the first line of input from the terminal after the user has logged on. In the example, it executes a CLIST or a REXX named BRDCST.

**DYNAMNBR** Defines the number of data sets that can be dynamically allocated at the same time. A constant of 2 is always added to the DYNAMNBR value you specify. It allows data sets to be more quickly reallocated because control blocks for data sets remain in storage, even after the data sets have been de-allocated. You should choose the value for DYNAMNBR carefully. The value should be large enough that it is not readily exceeded by the number of dynamic allocation requests made during the user's session. However, the larger the value you specify for DYNAMNBR the more virtual storage is used. The actual amount of virtual storage depends on the number of data sets the user allocates and de-allocates in a session. The value cannot exceed the number of concurrently allocated resources specified in the SYS1.PARMLIB member ALLOCxx, parameter TIOT SIZE. For details, refer to *z/OS MVS Initialization and Tuning Reference*, SA22-7592.

**SYSIN** Specifies that SYSIN is the user's terminal.

**SYSPRINT** Specifies that SYSPRINT is to be directed to the user's terminal.

Additional data sets can be allocated dynamically during the user's session or can be defined in the logon procedure. The following DD statements have special meaning and can be included in the logon procedure:

**SYSPROC** Defines the current REXX exec or CLIST library to be searched when the user uses the implicit form of the **EXEC** command. Figure 3-6 shows the explicit form of the **EXEC** command and the library name is specified in the command. The implicit form would be BRDCST.

**SYSEXEC** Defines the current REXX exec library concatenation to the **EXEC** command when users use the implicit form of the command. By default, the system searches SYSEXEC first, followed by SYSPROC.

The data sets described in SYSPROC and SYSEXEC DD statements must be partitioned, and have a record format of V, VB, F, or FB. You can allocate them dynamically using the **ALLOCATE** command and activate them with the **ALTLIB** command.

### 3.7 TSO/E logon process in a VTAM environment

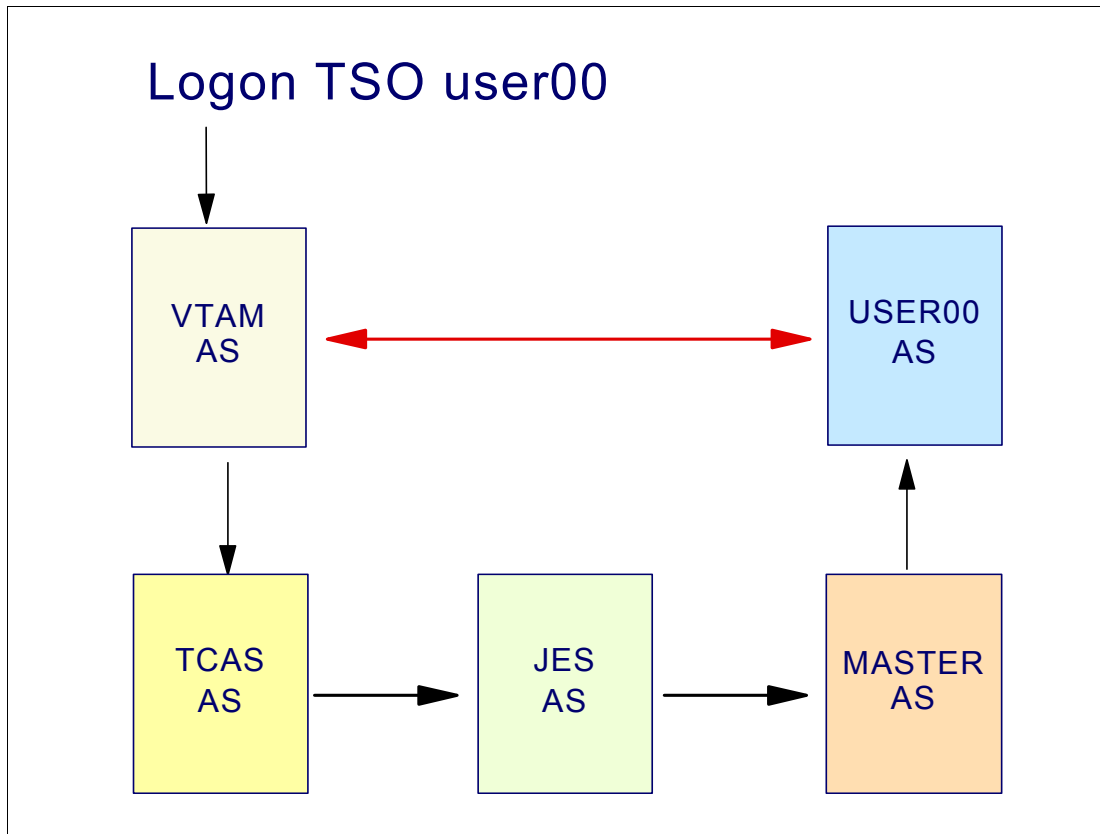


Figure 3-7 TSO/E logon process in a VTAM environment

In a VTAM environment, when a user enters a **LOGON** command to the TSO applid:

1. VTAM receives the command and passes it to the TCAS address space.
2. If the maximum number of users logged on in the system is reached, the logon is rejected; if not, and the user ID was not specified, TCAS prompts for the user ID.
3. Once the user ID is specified, TCAS verifies that the user has authority to use TSO/E. Depending on the installation customization, a full-screen logon panel is shown to the user. Figure 3-8 on page 93 shows the panel displayed when the user is RACF defined. The values shown in the fields PROCEDURE, ACCT NMBR, SIZE, and COMMAND are the same the user entered for the previous TSO/E session. If this is the first session, they are the default values. The command entered in the COMMAND field is executed after any command entered in the PARM field on the EXEC statement of the logon procedure.
4. After the Enter key is pressed, TSO/E verifies the values entered, then the user ID and the logon procedure name is passed to JES. The JCL is interpreted and converted. The MASTER creates the user address space and the resources specified in the JCL are allocated.
5. The user receives a screen with the **READY** prompt at the left top corner of the screen. This is called *line-mode TSO/E*. Now TSO/E is ready to accept commands, and user interfaces such as ISPF or SDSF can be called.



## 3.8 TSO/E full-screen logon panel

```
----- TSO/E LOGON -----

Enter LOGON parameters below:                RACF LOGON parameters:

Userid   ==> MIRIAM
Password ==> _
Procedure ==> IKJACCNT                       New Password ==>
Acct Nbr ==> ACCNT#                          Group Ident  ==>
Size     ==> 860000
Perform  ==>
Command  ==> ISPPDF

Enter an 'S' before each option desired below:
      -Nomail      -Nonotice      -Reconnect      -OIDcard

PF1/PF13 ==> Help   PF3/PF15 ==> Logoff   PA1 ==> Attention   PA2 ==> Reshow
You may request specific help information by entering a '?' in any entry field
```

Figure 3-8 TSO/E full-screen logon panel

To log on to TSO/E, type **LOGON** *yourid* and press the Enter key.

After you log on you receive a screen similar to the one shown in Figure 3-8, which is called a *panel*. A panel is a predefined display image that fills your screen. Notice that your user ID appears in capital letters to the right of the Userid arrow and other information required by your installation appears in capital letters to the right of other arrows. The computer generally re-displays information in uppercase regardless of how you typed it.

The area to the right of an arrow is called an *input field*. You can type information only in input fields. If you type anywhere else on the screen, the keyboard locks. To unlock the keyboard, press the Reset key.

In the Password input field you must enter the password that your administrator gave you. The characters do not appear on the screen when you enter your password. Suppressing the password in this way prevents others from seeing it as you type it.

Optionally, you can supply a new password (if you wanted to change it), a logon procedure if you do not want to use the installation default, an account number if required by the installation, a RACF group, a identification (that you are connect to), the region size if you need more than the installation default, and the first command to be executed after your userid is logged on. Figure 3-8 shows that a command **ISPPDF** was specified. In this example, ISPPDF is an installation CLIST that allocates the required data sets and calls ISPF. In such cases, instead of entering TSO/E in line-mode, the user would receive the ISPF Primary Menu panel and would be in full-screen mode.

## 3.9 TSO/E line-mode

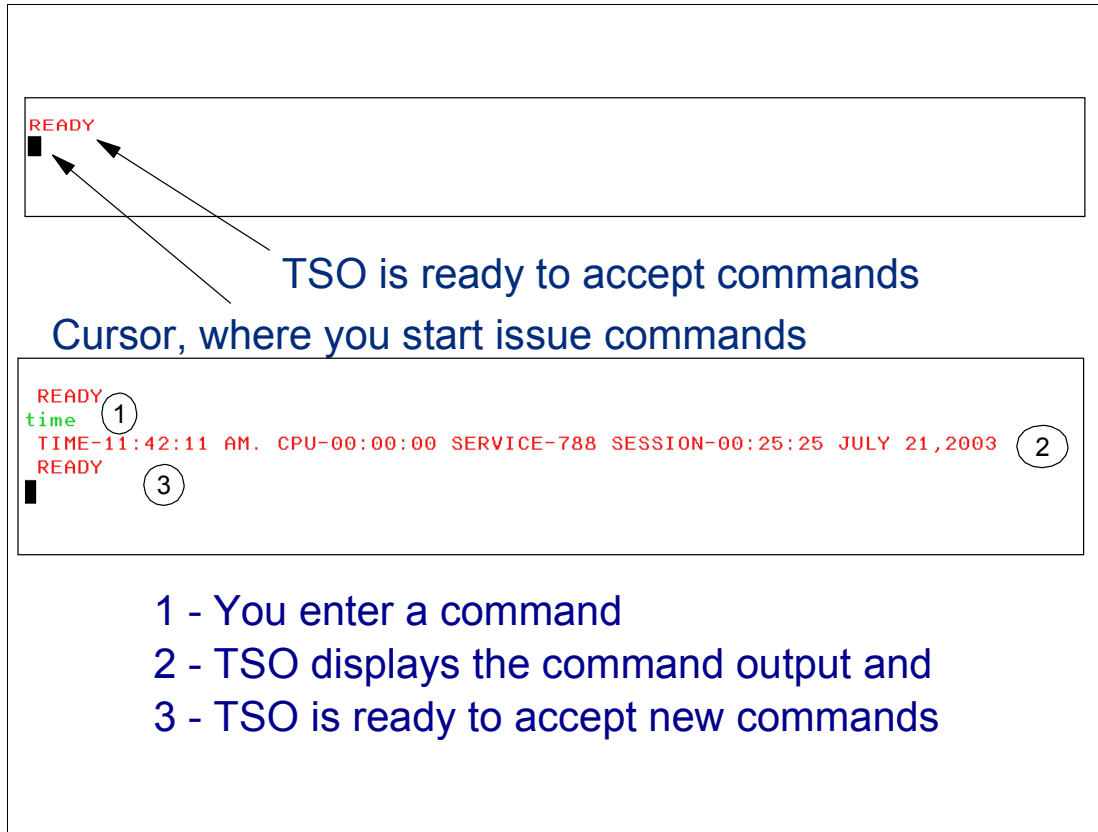


Figure 3-9 TSO/E line-mode

When you do not enter a command name in the panel shown in Figure 3-8, you enter TSO/E in line-mode. When you log on, you receive the screen shown in Figure 3-9. The word `READY` in the corner indicates that TSO is ready to accept your commands.

In TSO/E line-mode you type TSO/E commands one line at a time. It is a quick and direct way to use TSO/E and was the way programmers originally used to communicate interactively with the z/OS operating system.

You probably will not use TSO/E in line-mode. The user interface provided by ISPF is a more friendly way to work with TSO/E. In the following sections we present some hints to help you when you are using TSO/E and ISPF.

For more information, refer to *z/OS V1R1.0 TSO/E Primer*, SA22-7787, and *z/OS V1R3.0 TSO/E User's Guide*, SA22-7794.

### Interrupting a TSO/E function

The *Attention Interrupt* key allows you to interrupt or end a process that is taking place. If you are in a process and you want to stop or see a message requesting information you do not have, you can press the Attention Interrupt key to end the process.

The Attention Interrupt key often is labeled PA1. Sometimes it is called an *escape* key and is labeled Esc.

## 3.10 Using TSO/E as batch job

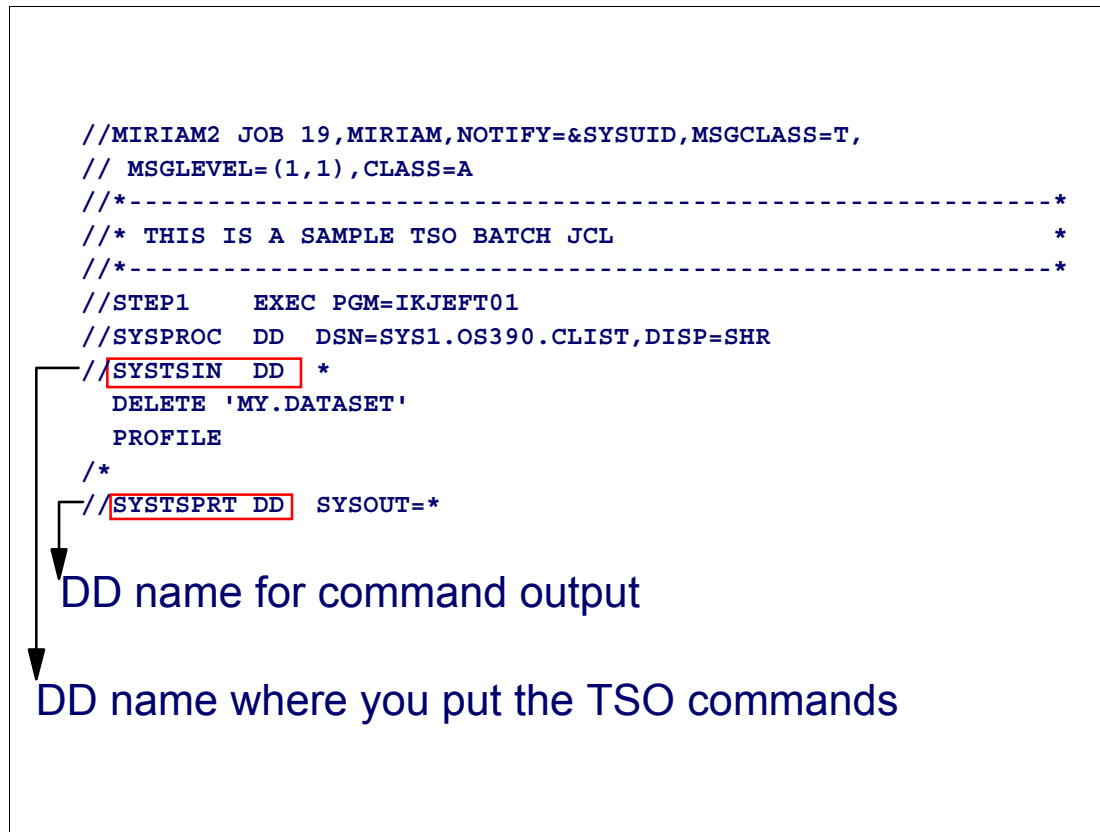


Figure 3-10 TSO/E batch

Instead of waiting at a terminal for your job to run, you can use the terminal to prepare a job containing the commands and data you would have entered at the terminal, then use the **SUBMIT** command to run the job. In this case, you are using the facilities of TSO/E exactly as if you submitted the commands individually at the terminal. You need the following JCL statements to submit your job:

- ▶ A JOB statement to identify your job
- ▶ An EXEC statement with the name of the TSO/E terminal monitor program (IKJEFT01, IKJEFT1A, or IKJEFT1B)
- ▶ At least, the following ddnames:
  - a. SYSTSPRT, which is used to control the output for your job. You can specify this DD as SYSOUT.
  - b. SYSTSIN, which is used as input for your TSO/E commands. It can be in stream (use a /\* to indicate the end of the stream).
- ▶ The DDNAMEs required by the application you intend to run.

## 3.11 TSO/E PROFILE command

```
READY
profile
CHAR(0) LINE(0) PROMPT INTERCOM NOPAUSE NOMSGID MODE WTPMSG NORECO
VER PREFIX(MIRIAM) PLANGUAGE(ENU) SLANGUAGE(ENU)
DEFAULT LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS TERMINAL ①
READY
delete old.dataset.sample1
ENTRY (A) MIRIAM.OLD.DATASET.SAMPLE1 DELETED ②
READY
profile noprefix ③
READY
profile
CHAR(0) LINE(0) PROMPT INTERCOM NOPAUSE NOMSGID MODE WTPMSG NORECO
VER NOPREFIX ④ PLANGUAGE(ENU) SLANGUAGE(ENU)
DEFAULT LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS TERMINAL
READY
delete old.dataset.sample2
ENTRY (A) OLD.DATASET.SAMPLE2 DELETED ⑤
READY
profile msgid list ⑥
IKJ56688I CHAR(0) LINE(0) PROMPT INTERCOM NOPAUSE MSGID MODE WTPMS
G NORECOVER NOPREFIX PLANGUAGE(ENU) SLANGUAGE(ENU)
IKJ56689I DEFAULT LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS TERMINAL
READY
profile prefix(newpref) list
IKJ56688I CHAR(0) LINE(0) PROMPT INTERCOM NOPAUSE MSGID MODE WTPMS
G NORECOVER PREFIX(NEWPREF) PLANGUAGE(ENU) SLANGUAGE(ENU)
IKJ56689I DEFAULT LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS TERMINAL
READY
```

Figure 3-11 TSO/E PROFILE command

Under TSO/E you always have a TSO profile, which is kept from one session to another. The way you issue the command to see your TSO profile depends on which environment you are running in:

- ▶ In TSO/E line-mode environment, type **PROFILE** and press the Enter key.
- ▶ In ISPF/PDF environment, type **TSO PROFILE** in the command line.

To change the TSO profile, type the **PROFILE** command followed by its operands and values. **PROFILE** can be shortened to **PROF**.

Figure 3-11 shows the command in line-mode. The command output is shown at 1. When you use **PREFIX** in your TSO profile, all data sets you refer to, when *not imbedded* in single quotes, are prefixed with your prefix, as shown at 2.

To inactivate the prefix, enter the command **PROFILE NOPREFIX**, as shown at 3. Now, you do not have to use quotes, but you have to inform the complete data set name, as shown at 5; otherwise, you will receive an error message or delete a wrong data set.

You can issue **PROFILE MSGID** to have all diagnostic messages you receive identified by their IDs, as shown at 6.

You can add the **LIST** operand in your **PROFILE** command to list the new profile after the change, as shown at 6.

For more information about the **PROFILE** command and its operands, refer to *z/OS TSO/E Command Reference*, SA22-7782.

## 3.12 TSO/E languages

<p><b>REXX</b></p> <pre>/*REXX */ parse upper arg ax Address 'ISPEXEC' "LIBDEF ISPLLIB DATASET ID('VAINI.U.PANELS')" if rc = 0 then do   Address "TSO"   aplst = "'VAINI.U.CLIST'"   if aplst ^= "" then ,   "ALTLIB ACT APPL(CLIST) DS("aplst)"   Address "ISPEXEC"   "SELECT CMD(DR" ax ") NEWAPPL(DREQ) PASSLIB "   Address "TSO"   if aplst ^= "" then ,   "ALTLIB DEACT APPL(CLIST)"   Address 'ISPEXEC'   "LIBDEF ISPLLIB " end else say '*** ERROR*** DR must be run under ISPF' Exit 0</pre>	<p><b>CLIST</b></p> <pre>PROC 0 DB IF .&amp;DB = .DB THEN +   CONTROL LIST SYMLIST CONLIST MSG PROMPT CONCATD F(SYSPROC) DA('DB2V710S.NEW.SDSNCLST') BEFORE CONCATD F(ISPLLIB) DA('DSN710.QPP.RUNLIB.LOAD') BEFORE ISPEXEC SELECT CMD(DSNECPRI SSID(DB2S)) NEWAPPL(DSNE) DECON F(ISPLLIB) DA('DSN710.QPP.RUNLIB.LOAD') DECON F(SYSPROC) DA('DB2V710S.NEW.SDSNCLST')</pre>
--	---

Figure 3-12 TSO/E languages

There are two languages available in the TSO/E environment: CLIST and REXX.

CLIST is an interpretative language that helps you to work more efficiently with TSO/E. It is a command list language because the most basic CLISTs are lists of TSO/E commands. When invoked, it issues the TSO/E commands in sequence. The CLIST language includes the programming tools you need to write extensive, structured applications. CLISTs can perform a number of complex tasks, from displaying a series of full-screen panels to managing programs written in other languages.

The REstructured eXtended eXecutor (REXX) language is a programming language that is extremely versatile. Aspects such as common programming structure, readability, and free format make it a good language for beginners and general users. Yet because the REXX language can be intermixed with commands to different host environments, provides powerful functions, and has extensive mathematical capabilities, it is also suitable for more experienced computer professionals. The TSO/E implementation of the REXX language allows REXX execs to run in any MVS address space. You can write a REXX exec that includes TSO/E services and run it in a TSO/E address space, or you can write an application in REXX to run outside of a TSO/E address space.

CLIST and REXX can be used to customize and tailor your TSO/E environment specifically for the applications you want to use. Figure 3-12 shows a sample CLIST procedure and a REXX exec.

### 3.13 Interactive System Productivity Facility (ISPF)

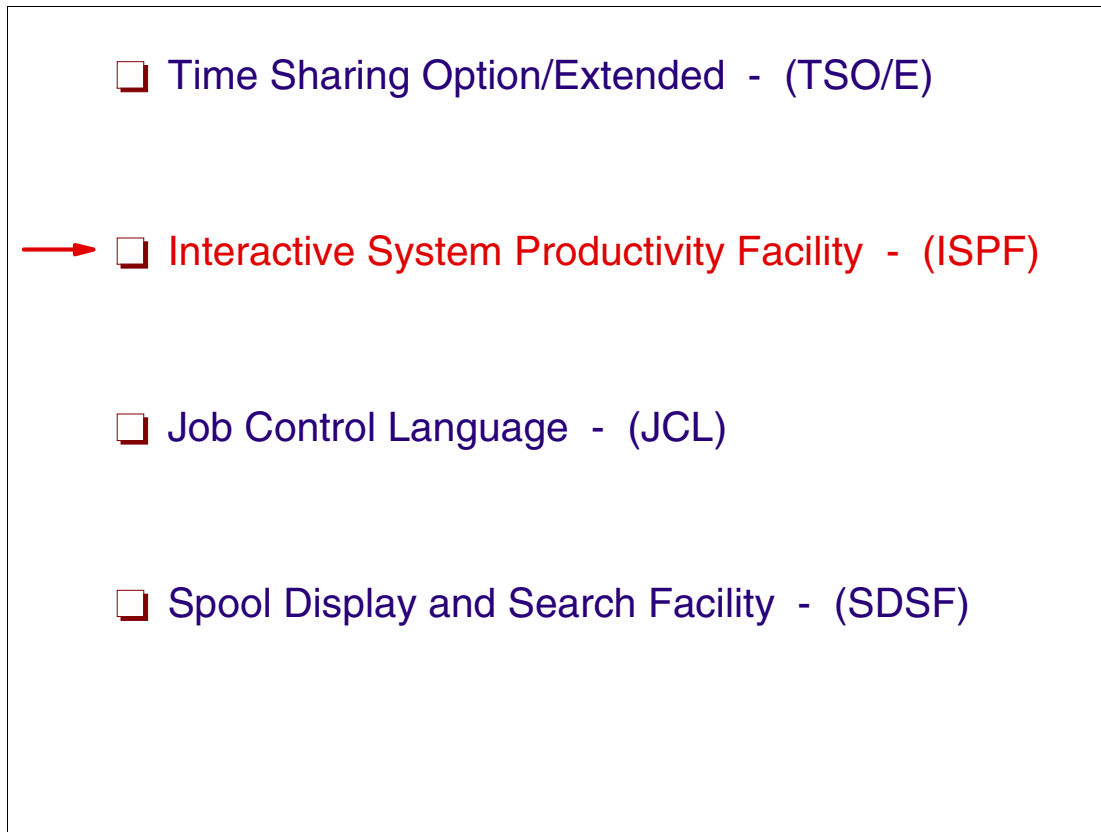


Figure 3-13 Interactive System Productivity Facility (ISPF)

The Interactive System Productivity Facility/Program Development Facility (ISPF/PDF) is a set of panels that help you manage libraries of information on the MVS system. The libraries are made up of units called *data sets* that can be stored and retrieved. You can have different kinds of information in data sets. Some examples are:

- ▶ Source code
- ▶ Data such as inventory records, personnel files, or a series of numbers to be processed
- ▶ Load modules

ISPF can be used in many ways. Some examples are the following:

- ▶ Users can edit, browse and print data.
- ▶ Data processing administrators and system programmers can use ISPF to:
  - Monitor and control program libraries
  - Communicate with MVS through TSO commands, CLISTs, or REXX EXECs
- ▶ Programmers can use ISPF to develop a batch, interactive, or any other type of program and its documentation.
- ▶ Terminal users can invoke a wide range of utilities like search, compare, compilers, and so forth.
- ▶ Programmers can use ISPF services to develop dialogs.

### 3.14 ISPF: Data set types supported

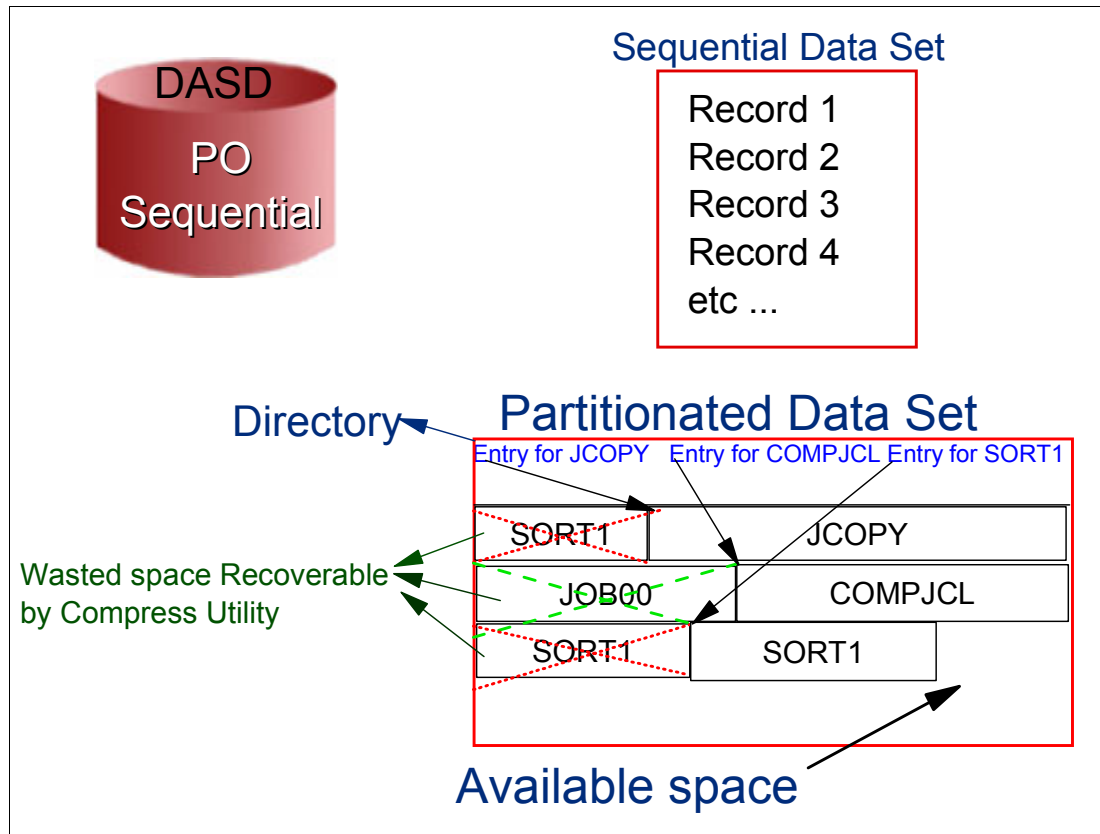


Figure 3-14 ISPF and ISPF/PDF data set types supported

A data set is a collection of logically related data; it can be a source program, a library of macros, or a file of data records used by a processing program. *Data records* are the basic unit of information used by a processing program. ISPF supports the following data set types for any ISPF options, such as Edit, Browse, and Delete:

- ▶ Sequential
- ▶ Partitioned (PDS) and Partitioned Extended (PDSE)

For VSAM data sets, ISPF supports creation, obtaining data set information, and deletion. Browse and Update is supported by DITTO, an IBM licensed program installed under ISPF.

ISPF does not support:

- ▶ Record format variable block spanned (VBS) data sets
- ▶ Direct Access (DA) data sets
- ▶ Tape data sets
- ▶ Generation data group (GDG) base data sets

In a sequential data set, records are stored and retrieved in a sequential order.

A *partitioned data set* is like a collection of sequential data sets, called members, each one having a name. A directory index is used to locate members in the partitioned data set. The directory consists of 256-byte records, each one containing directory entries. There is one directory entry for each member.

## 3.15 ISPF: Data set and member naming conventions

- ❑ Data Set Name Convention:
  - Unique name
    - Maximum 44 Characters
  - Maximum of 22 name segments: level qualifier
    - The first name in the left: High Level Qualifier (HLQ)
    - The last name in the right: Low Level Qualifier (LLQ)
    - Level qualifiers are separated by '.'
  - Each level qualifier:
    - From 1 up to 8 characters
    - The first must be alphabetical (A-Z) or national (@ # \$)
    - The 7 remaining: alphabetical, national, numeric (0-9) or hyphen (-)
  - Example: MYID.JCL.FILE2 HLQ: MYID 3 qualifiers
- ❑ Member Name of Partitionated Data Set:
  - 8 Bytes
  - First byte: alphabetical (A-Z) or national (@ # \$)
  - The 7 remaining: alphabetical, national, numeric (0-9)

Figure 3-15 ISPF: Data set and member naming conventions

ISPF allows you to create data sets and member names that follow the ISPF naming convention shown in Figure 3-15.

All data sets and member names created within ISPF are converted to uppercase. If you create members outside of ISPF that do not meet these conventions, they are displayed in ISPF member lists and can be selected from those lists. These member names can also be specified for the Browse service, with the exception of member names containing lowercase alphabetic characters. (ISPF converts the member name to uppercase before searching for the member and therefore cannot process a lowercase member.) Member names not meeting the ISPF naming convention are not supported for the other ISPF services. The same applies to data sets.

ISPF has very useful online help and tutorial facilities (available while using ISPF). For example, if you need help filling in the data requested by an ISPF utility, you can use the tutorial to help you understand the data entry requirements for that utility. For information about using the tutorial, see *z/OS V1R2.0 ISPF Dialog Developer's Guide*, SC34-4821.



## 3.16 ISPF components

- ❑ Dialog Manager
  - Functions
    - REXX or CLIST
    - Programs
  - Panels
  - Messages
  - Tables
  - Skeletons
  - Dialog variables
- ❑ PDF
- ❑ SCLM
- ❑ Client/Server - The Workstation Agent Component

Figure 3-16 ISPF components

ISPF consists of four major components as shown in Figure 3-16. These components are considered one element in all releases of z/OS.

### ISPF Dialog Manager (DM)

Dialog Manager provides services to dialogs and end-users. A *dialog* is the interaction between a person and a computer. It helps a person who is using an interactive display terminal to exchange information with a computer. The user starts an interactive application through an interface that the system provides. The dialog with the user begins with the computer displaying a panel and asking for user interaction. It ends when the task for which the interactions were initiated is completed.

Dialog Manager is composed of six elements:

- ▶ Functions: A function is a command procedure or a program that performs processing requested by the user. It can invoke ISPF dialog services to display panels and messages, build and maintain tables, generate output data sets, and control operational modes. They can be written as:
  - REXX or CLIST command procedures
  - Programs
- ▶ Panel definitions: A panel definition is a programmed description of the panel. It defines both the content and format of a panel. Most panels prompt the user for input. The user's response can identify which path is to be taken through the dialog, as on a selection panel. The response can be interpreted as data, as on a data-entry panel.

- ▶ **Message definitions:** Message definitions specify the format and text of messages to users. A message can confirm that a user-requested action is in progress or completed, or it can report an error in the user's input.
- ▶ **Table:** Tables are two-dimensional arrays that contain data and are created by dialog processing. They can be created as a temporary data repository, or they can be retained across sessions. A retained table can also be shared among several applications. The type and amount of data stored in a table depends on the nature of the application. Not all dialogs use tables.
- ▶ **File tailoring skeletons:** Skeletons work like a fill-in-the-blank exercise. They take dialog variables and put them into a data set containing statements that control the output format. The output data set can be used to drive other processes. File skeletons are frequently used to produce job data sets for batch execution. Some dialogs can use this kind of resource.
- ▶ **Dialog variables:** ISPF services use variables to communicate information among the various elements of a dialog application. ISPF provides a group of services for variable management. Variables can vary in length from zero to 32K bytes.

### **Program Development Facility (ISPF/PDF)**

Program Development Facility provides View, Browse, Edit, and library access services that can be combined in a dialog with any of the ISPF services. The library access services carry out functions involving members of a programming library. These functions include adding, finding, and deleting members, and displaying member lists.

### **Software Configuration Library Manager (SCLM)**

Software Configuration Library Manager (SCLM) is a software tool that helps you develop complex software applications. Throughout the development cycle, SCLM automatically controls, maintains, and tracks all of the software components of the application. You can lock the version being edited in a private library and then promote it. Use SCLM to create, control, maintain, and track software components for a project. For more information about SCLM, refer to *z/OS V1R2.0 ISPF Software Configuration Library Manager Reference*, SC34-4818.

### **ISPF Client/Server - The Workstation Agent Component**

The Workstation Agent Component enables you to run ISPF on a programmable workstation and display the panels using the display function of your workstation operating system. Manuals in the ISPF library refer to this as running in GUI mode. The ISPF WSA is supported on the following platforms:

- ▶ OS/2®
- ▶ Microsoft® Windows
- ▶ AIX®
- ▶ HP-UX
- ▶ Solaris

Connecting to a workstation for data access has a direct impact on your installation's CPU processing time. One reason for using the ISPF Client/Server function is to offload CPU cycles from the host to a less expensive workstation. But even if that is not your goal, an added benefit is that your users can use the connection for distributed editing. This means that they can use their favorite editor to work with your data, whether that means using a host editor on host and workstation files, or using a workstation editor on the same files. By making the connection to the workstation, a user can edit workstation files on ISPF, or host files on his workstation. The distributed edit function can be used in standard 3270 mode, or in ISPF GUI mode.

## 3.17 Sample CLIST to allocate ISPF and SDSF data sets

```
ALLOC FI(SYSEXEC) DS('ISP.SISPEXEC' 'ISF.SISFEXEC') REUS SHR
ALLOC FI(SYSPROC) DS('USER.CLISTS.DSN' 'ISP.SISPCLIB') REUS SHR
ALLOC FI(ISPPLIB) DS('ISP.SISPSENU' 'ISF.SISFPLIB') SHR
ALLOC FI(ISPMLIB) DS('ISP.SISPMENU' 'ISF.SISFMLIB') SHR
ALLOC FI(ISPSLIB) DS('ISP.SISPSENU' ) SHR REUS
ALLOC FI(ISPTLIB) DS('ISP.SISPTENU' 'ISF.SISFTLIB') SHR REUS
ALLOC FI(SYSHELP) DS('ISP.SISPHELP' 'ISF.SISFTLIB') SHR REUS
ALLOC FI(ISPTABL) DS('USERID.PROFILE.DSN') SHR
ALLOC FI(ISPPROF) DS('USERID.PROFILE.DSN') SHR
ALLOC FI(ISPLIST) SYSOUT(A) LRECL(121) RECFM(F B A)
ALLOC FI(ISPLOG) SYSOUT(A) LRECL(121) RECFM(V A)
ALLOC FI(ISPCTL0) NEW UNIT(VIO) SPACE(1,1) CYLINDERS LRECL(80) +
    BLKSIZE(800) RECFM(F B)
ALLOC FI(ISPCTL1) NEW UNIT(VIO) SPACE(1,1) CYLINDERS LRECL(80) +
    BLKSIZE(800) RECFM(F B)
ALLOC FI(ISPCTL2) NEW UNIT(VIO) SPACE(1,1) CYLINDERS LRECL(80) +
    BLKSIZE(800) RECFM(F B)
ALLOC FI(ISPWRK1) NEW UNIT(VIO) SPACE(1,1) CYLINDERS LRECL(256) +
    BLKSIZE(2560) RECFM(F B)
ALLOC FI(ISPWRK2) NEW UNIT(VIO) SPACE(1,1) CYLINDERS LRECL(256) +
    BLKSIZE(2560) RECFM(F B)
ALLOC FI(ISPLST1) NEW UNIT(VIO) SPACE(1,1) CYLINDERS LRECL(121) +
    BLKSIZE(1260) RECFM(F B A)
ALLOC FI(ISPWRK2) NEW UNIT(VIO) SPACE(1,1) CYLINDERS LRECL(121) +
    BLKSIZE(1260) RECFM(F B A)
ISPSTART PANEL(ISR@PRIM) NEWAPPL(ISR)
```

Figure 3-17 Sample CLIST to allocate ISPF and SDSF data sets

ISPF runs in a TSO/E environment. Before ISPF can be started, some data sets must be available; this can be done in any of the following ways:

1. Allocate them in the logon procedure
2. Dynamic allocation using the TSO/E **ALLOC** command, as shown in Figure 3-21
3. Save all commands shown in Figure 3-17 as a command and invoke during the logon process as the **ISPPDF** command shown in Figure 3-8 on page 93.

The required DD names are:

- ▶ SYSEXEC for partitioned data sets containing REXX execs
- ▶ SYSPROC for partitioned data sets containing CLIST and/or REXX execs
- ▶ ISPPLIB for panel definitions libraries
- ▶ ISPTLIB for input table definitions libraries
- ▶ ISPMLIB for message libraries
- ▶ ISPPROF for the *user profile* data set under ISPF/PDF environment. ISPF/PDF use this data set for storing variables and settings to be used from one TSO/ISPF session to another.

The DD names ISPTABL (for output tables) and ISPSLIB (for skeletons) may be requested for some specific applications.

For some data sets, if not pre-allocated, ISPF allocates them:

- ▶ ISPLOG DD name: The data set where ISPF logs commands issued by the user and some ISPF functions log errors detected.
- ▶ ISPLIST DD name: ISPF uses this data set when user requests printed output.

The DD names ISPCTLx (where x can be 1–9, A–W) are used by ISPF as a temporary data set. ISPF can use one for each logical screen to generate JCL or utility control statements or to generate listings. ISPF can run up to 32 logical screens at one time. The default value is 8. The installation can change the default value by modifying the ISRCONFIG table. ISPCTL0 is used only by Edit for the **SUBMIT** command.

The DD names ISPWRKx are used by ISPF for file tailoring services with ISPFILe allocated to a PDS. The DD names ISPLSTx are used for generated listings. The same pre-allocation can be done by the **ALLOCATE** command in a CLIST or in a REXX exec to be executed before the ISPF start.

## 3.18 ISPF primary option menu

```
Menu Utilities Compilers Options Status Help
-----
ISPF Primary Option Menu
Option ==>
0 Settings      Terminal and user parameters      User ID . : MIRIAM
1 View          Display source data or listings   Time. . . : 16:56
2 Edit          Create or change source data      Terminal. : 3278
3 Utilities     Perform utility functions         Screen. . : 1
4 Foreground    Interactive language processing   Language. : ENGLISH
5 Batch         Submit job for language processing Appl ID . : PDF
6 Command       Enter TSO or Workstation commands TSO logon : IKJACCT
7 Dialog Test   Perform dialog testing           TSO prefix: NEWPREF
8 LM Facility   Library administrator functions   System ID : SC43
9 IBM Products  IBM program development products  MVS acct. : ACCNTH
10 SCLM         SW Configuration Library Manager  Release . : ISPF 5.2
11 Workplace    ISPF Object/Action Workplace

Enter X to Terminate using log/list defaults

F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward    F9=Swap
F10=Actions  F12=Cancel
```

Figure 3-18 ISPF primary option menu

ISPF is started in a TSO/E environment through an ISPF, or PDF, or ISPSTART command. The ISPF Primary Option Menu contains the options that you can use to create your own applications online. If your installation has a customized ISPF Primary Option Menu, the menu might not contain all of options shown in Figure 3-18, or it might contain certain installation-specific options.

Most ISPF panels have action bars at the top; many panels also have point-and-shoot text fields.

## 3.19 ISPF panel areas

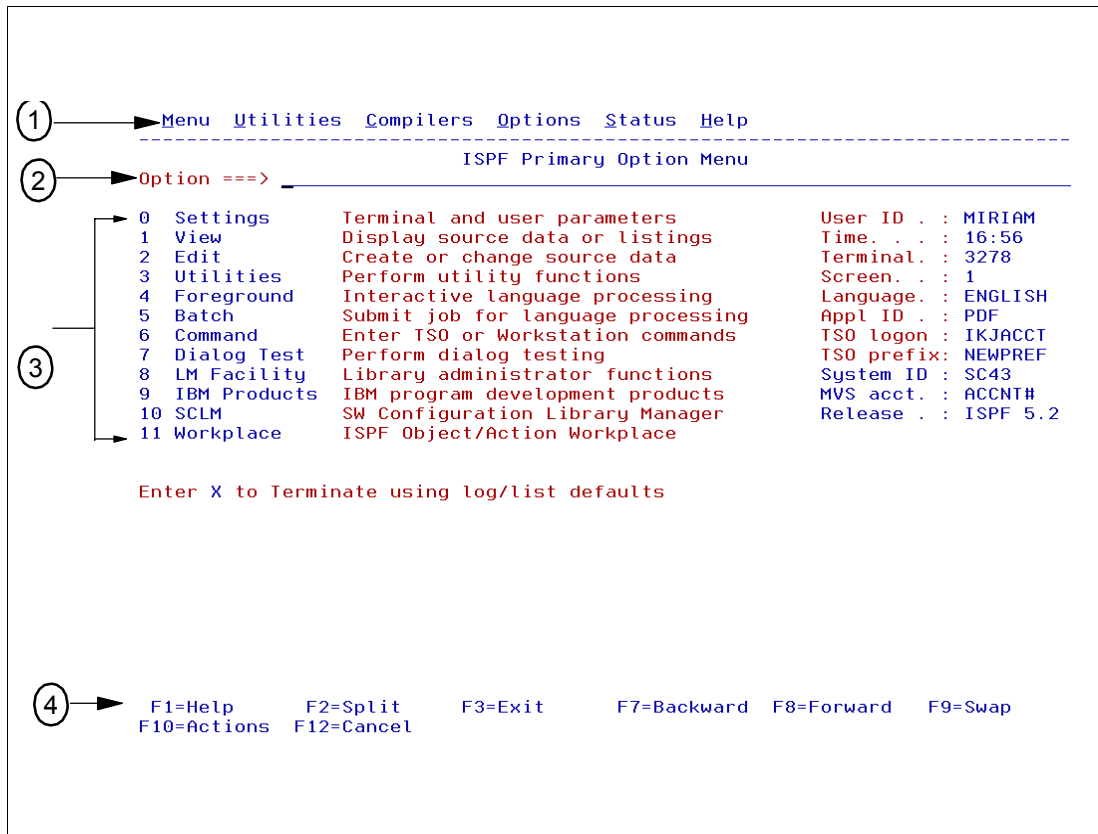


Figure 3-19 ISPF panel areas

ISPF panels can contain the following areas:

1. Action bar: The area at the top of an ISPF panel that contains choices that give you access to actions available on that panel.
2. Option: The fields in this column are point-and-shoot text fields. For example, if you want to select the Edit Panel, to create or change source data, you type it in the option input field and press Enter. You can also type commands in the Option field.
3. Dynamic status area: You can specify what you want to be displayed in this area; in Figure 3-19 it is descriptions of all the possible options you can take from this panel.
4. Function Keys: ISPF uses Common Use Access (CUA®)-compliant definitions for function keys F1-F12. For example: PF1 is used to invoke the ISPF Help Function. You can eliminate this area by typing **PFSHOW OFF** in area 2, the Option field.

## 3.20 Action bars

```

Menu  Utilities  Compilers  Options  Status  Help
-----
Optio  ─ 1. Library          ary Option Menu
      2. Data set
      3. Move/Copy
0 Se   4. Data Set List   arameters
1 Vi   5. Reset Statistics or listings
2 Ed   6. Hardcopy      urce data
3 Ut   7. ISPF C/S Install... ctions
4 Fo   8. Outlist       e processing
5 Ba   9. Commands...  uage processing
6 Co  *0. Reserved      ation commands
7 Di  11. Format        ing
8 LM  12. SuperC       or functions
9 IB  13. SuperCE      ment products
10 SC 14. Search-For   brary Manager
11 Wo 15. Search-ForE  Workplace

Enter X to Terminate using log/list defaults

F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward  F9=Swap
F10=Actions  F12=Cancel

```

Figure 3-20 Action bars

Action bars give you another way to move through ISPF. If the cursor is located somewhere on the panel, there are several ways to move it to the action bar:

- ▶ Use the cursor movement keys to manually place the cursor on an action bar choice.
- ▶ Type ACTIONS on the command line and press Enter to move the cursor to the first action bar choice.
- ▶ Press F10 (Actions) or the Home key to move the cursor to the first action bar choice.

**Note:** ISPF does not provide a mouse emulator program. You can use Select in conjunction with point-and-shoot text fields and action bar choices to simulate moving the cursor to a field and pressing Enter.

When you make an action bar selection, the selected item is highlighted and ISPF displays a pull-down menu, that is, a list of numbered choices extending from the selection you made on the action bar. Figure 3-20 shows the result when you select Utilities on your screen.

You can select an action either by typing in its number and pressing Enter or by selecting the action with your cursor. ISPF displays the requested panel. If your choice contains an ellipsis (...), ISPF displays a pop-up window. When you exit this panel or pop-up, ISPF closes the pull-down menu and returns you to the panel from which you made the initial action bar selection.

## 3.21 Customizing your TSO/ISPF/PDF session

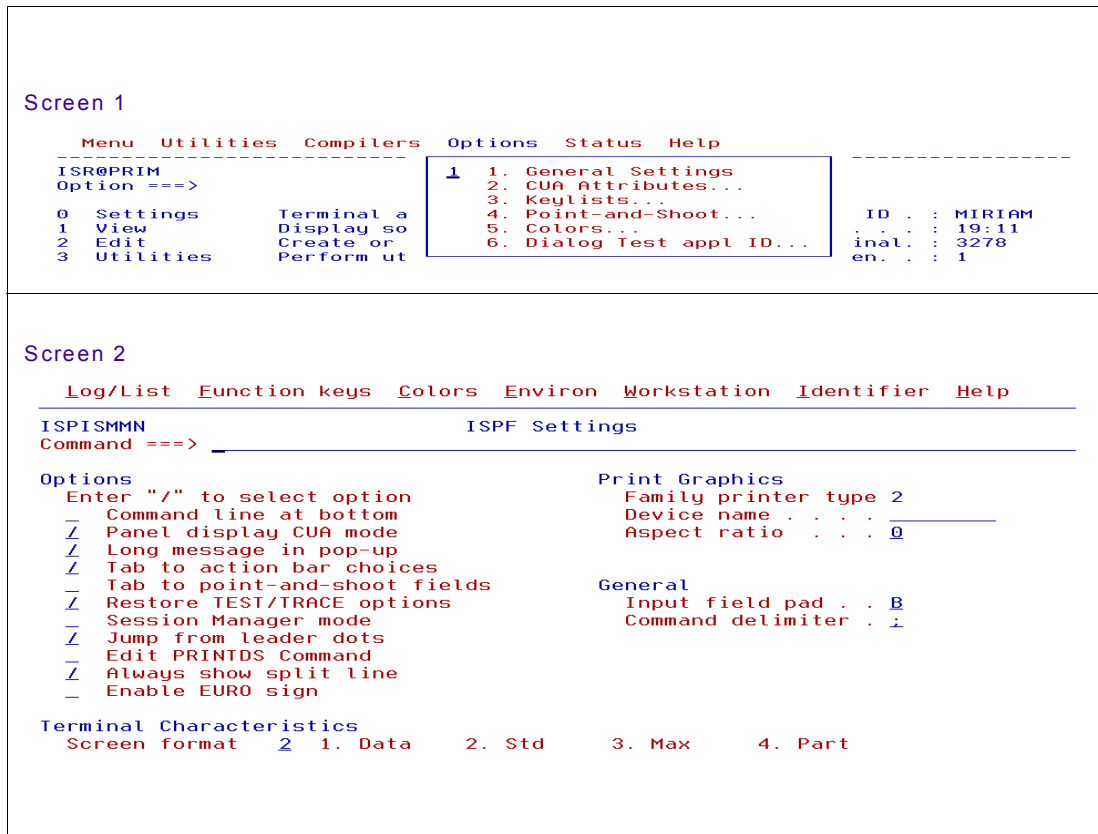


Figure 3-21 Customizing your TSO/ISPF/PDF session

You can customize your TSO/ISPF/PDF session by selecting **Options** → **General Settings** as shown in Figure 3-21, Screen 1. In the resulting screen, Screen 2, the first line shows others action fields where you can choose to change default settings, like colors, function keys, and so forth.

### ISPF Help function

ISPF has a powerful Help function which can teach you how to use all the default ISPF options. Access it by pressing the Help program function key or typing `help` in the Command line. You can see and change the program function key assignments by using the **KEYS** command.

You can learn how to use ISPF by just using the help function. At the Primary Option Menu press Help to learn about the options. In the Help panel, press Help again to learn how to move through help panels. Some Help options are cursor-sensitive, meaning you have to move the cursor to that option to get more information.

A good source of information for beginners in the ISPF environment is *Interactive System Productivity Facility Getting Started*, SC34-4440.



## 3.22 Allocating data sets: Utility option

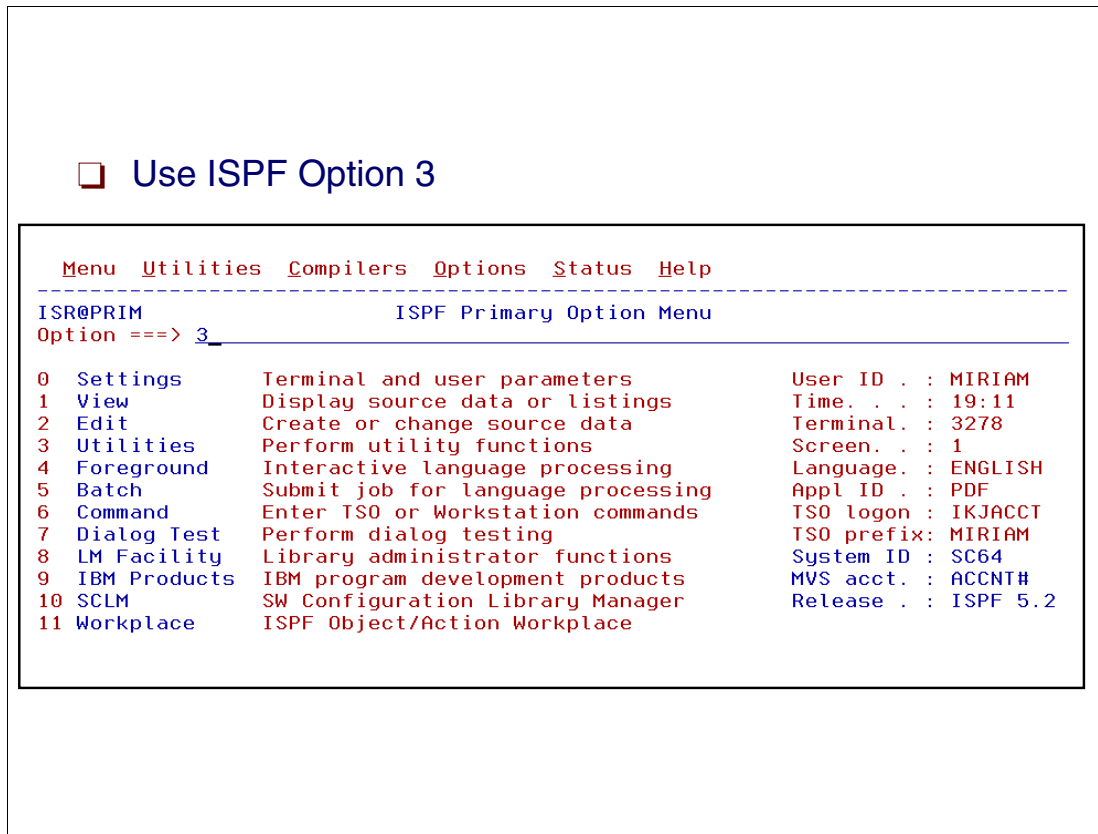


Figure 3-22 Allocating a data set: Utilities option

A data set, or file, is an area that is reserved on either disk or tape to enable you to write programs and store data. Before you can edit or store data, you must instruct the system to allocate some space on disk, often referred to as Direct Access Storage Devices (DASD), and provide information to identify the format of this data set.

There are two ways to allocate a data set:

1. Using ISPF dialogs
2. Using JCL

The first way is as follows:

In the primary menu, type **3** in the Option input field, and press the Enter key. Option 3 displays the utilities panel, as shown in Figure 3-23 on page 110.

## 3.23 Utility Selection Panel

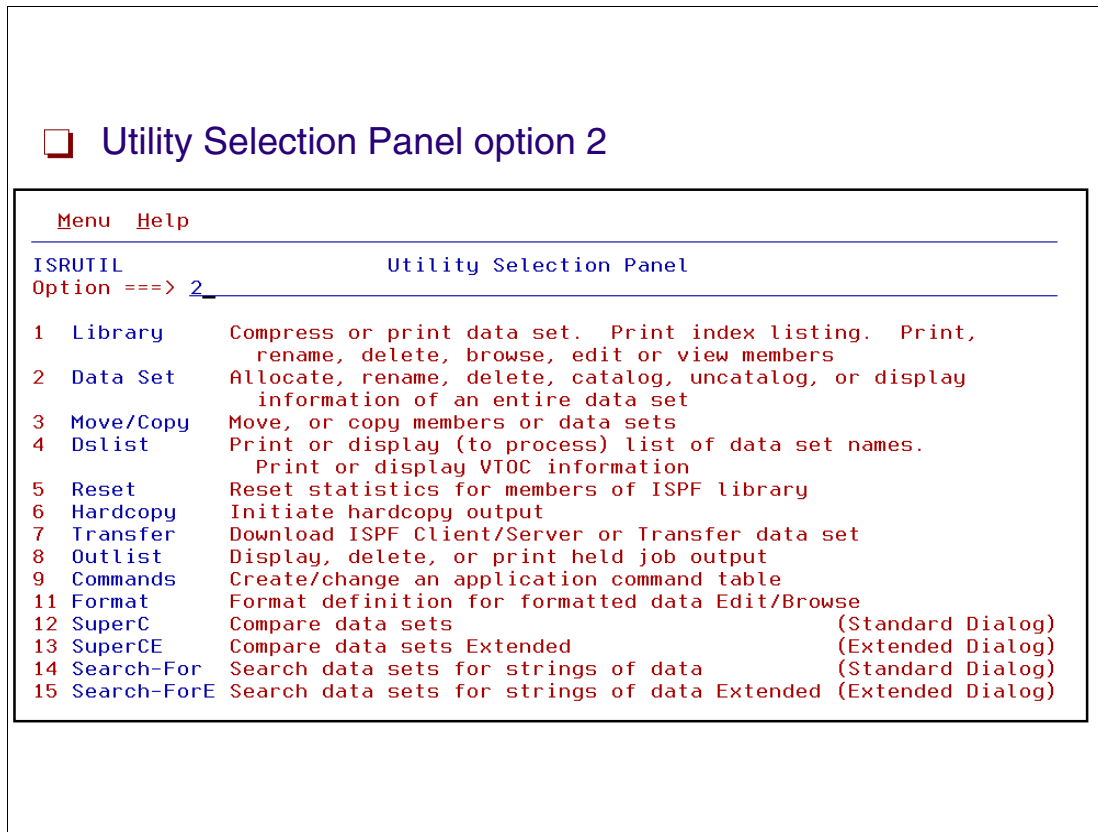


Figure 3-23 Utility Selection Panel option 2

The Utility Selection Panel gives you the ability to select utilities to perform a variety of functions, such as:

- ▶ Compress or print a data set
- ▶ Create, rename, delete a data set
- ▶ Move or copy data sets or members
- ▶ Search for strings in a data set
- ▶ Compare data sets

ISPF Option 3.2 enables you to reserve some space on a storage device and identify this space with a data set name, often referred to as a DSN or DSNAME.

In this example you will allocate a data set; to do this, type 2 in the Option field and press Enter.

## 3.24 Data Set Utility: Allocating a data set

```
Menu RefList Utilities Help
-----
ISRUDA2S                               Data Set Utility
Option ==> a_

  A Allocate new data set                C Catalog data set
  R Rename entire data set              U Uncatalog data set
  D Delete entire data set              S Short data set information
blank Data set information              V VSAM Utilities

ISPF Library:
Project . . . _____
Group . . . _____
Type . . . _____

Enter "/" to select option
/ Confirm Data Set Delete

Other Partitioned, Sequential or VSAM Data Set:
Data Set Name . . . _____
Volume Serial . . . _____ (If not cataloged, required for option "C")
Data Set Password . . . _____ (If password protected)

F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward  F9=Swap
F10=Actions  F12=Cancel
```

Figure 3-24 Allocating a data set - Data Set Utility panel

The Data Set Utility panel presents a variety of actions you can perform. You can allocate, delete, rename, catalog, uncatalog, or obtain information about a specific data set. The V option allows you create and obtain information about VSAM data sets in a very easy way.

From the ISPF Primary Option Menu you can go directly to this panel by typing =3.2 in the Option field.

To allocate a data set, type A on the Option field. Next you specify the name of data set you want to create:

- ▶ A data set with a three-level name can be entered in the Project, Group, and Type fields as shown in area 1 in Figure 3-24. Type one level of the name in each field. ISPF does not add your prefix as HLQ.
- ▶ You can specify the data set name and optionally a volume, as shown in area 2 in of the panel:

```
Other Partitioned, Sequential or VSAM Data Set:
Data Set Name. . . 'HLQ.XXX.XXXX.XXX'
```

**Note:** Imbed the data set name in single quotes unless you want the HLQ to equal your prefix.

If both a library (1) and a data set name (2) are specified on the same panel, the data set name takes priority. Therefore, to specify a library, leave the Data Set Name field blank. We

did not enter the data set name option field; therefore the values specified in the library name are used. See Figure 3-15 on page 100 for data set naming conventions.

Data set name standards are site-dependant, and may be protected by RACF or another security product. If you do not have the authority to allocate a DSN with a specific name, your request fails and you receive an error message.

Examples of some data set names:

- ▶ SYS1.PARMLIB
- ▶ SYS1.PARMLIB.BACKUP.D99156
- ▶ MIRIAM.PRIVATE.JCLLIB
- ▶ MIRIAM.TEST.NEW.SYSTEM.JCLLIB

The High Level Qualifier (HLQ) is the first part of the DSN. In the examples, the HLQs are SYS1 (usually reserved for MVS system DSNs), and MIRIAM, which could be your user ID. Some system data sets must be named as specified, but personal or in-house data sets should be named to be meaningful and easy to associate with a user or application, and to enable efficient security and DASD maintenance strategies to be maintained.

## 3.25 Allocate New Data Set panel

```

Menu  RefList  Utilities  Help
-----
Allocate New Data Set
Command ==> _____

Data Set Name . . . . : NEWPREF.MIRIAM.PRIVATE.JCLLIB

Management class . . . . _____ (Blank for default management class)
Storage class . . . . _____ (Blank for default storage class)
Volume serial . . . . _____ (Blank for system default volume) **
Device type . . . . sysda _____ (Generic unit or device address) **
Data class . . . . _____ (Blank for default data class)
Space units . . . . trks _____ (BLKS, TRKS, CYLS, KB, MB, BYTES
or RECORDS)
Average record unit _____ (M, K, or U)
Primary quantity . . . 90 _____ (In above units)
Secondary quantity . . 15 _____ (In above units)
Directory blocks . . . 100 _____ (Zero for sequential data set) *
Record format . . . . fb _____
Record length . . . . 80 _____
Block size . . . . 3120 _____
Data set name type : pds _____ (LIBRARY, HFS, PDS, or blank) *
(Y Y/MM/DD, YYYY/MM/DD
YY.DDD, YYYY.DDD in Julian form
Enter "/" to select option
DDDD for retention period in days
or blank)
_ Allocate Multiple Volumes

( * Specifying LIBRARY may override zero directory block)

( ** Only one of these fields may be specified)
F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward  F9=Swap
F10=Actions  F12=Cancel

```

Figure 3-25 Allocate New Data Set panel

After pressing the Enter key, the Allocate New Data Set panel is displayed. This panel shows the information you have to provide in order to allocate the new data set.

In this example we are allocating a partitioned data set (PDS); this type of data set allows individual members within the data set. Most environments now utilize DFSMS/MVS® to control data set allocation; therefore, it is not necessary to specify management class, storage class, or data class information. In most cases the defaults are satisfactory. We must, however, specify the following:

**Space units** The allocation units for our data set. The allocation units can be in blocks (BLKS), tracks (TRKS), cylinders (CYLS), KB, MB, bytes, or records.

**Primary quantity** The amount of primary space units we want to allocate.

**Secondary quantity** The space that can be allocated for secondary extents, if the primary quantity fills up. Non-extended data sets can have a maximum of 16 extents, which includes up to five multiple extents that may have been used to satisfy the primary extents.

**Note:** The exception to the 16 extent limitation is partitioned data set extended (PDSE) data sets, which can have up to 123 extents.

**Directory blocks** Must be specified for partitioned data sets. A directory is an index used to locate members in the partitioned data set. Each directory

record consists of 256 bytes containing directory entries. There is one directory entry for each member. The directory is written at the beginning of the primary space. It must fit in the first extent of the data set. For partitioned data sets, be sure to request enough directory space to allow for growth of the data set. You cannot lengthen the directory once the data set is created. If the directory runs out of space, you must recreate the data set.

**Note:** The number of member entries that fit in a directory block is as follows:

- For a data set with ISPF statistics: six entries per block
- For a data set without ISPF statistics: 21 entries per block
- For a load module data set: four to seven entries, depending upon attributes

**Record format**

Can be any valid combination of the following:

- F Fixed length records
- V Variable length records
- U Undefined format records
- B Blocked records
- A ASA printer control characters
- M Machine code printer control characters
- S Standard (for F) or spanned (for V) - sequential data sets only
- T Track-overflow feature

The option we used for a partitioned data set was **FB**.

**Record length**

The logical record length, in bytes, of the records to be stored in the data set. In the case of a JCL or program library this value is 80 bytes.

**Block size**

The block size (physical record length), in bytes, of the blocks to be stored in the data set. If records are specified, the block size specifies the average record length.

After pressing the PF3 (Exit) key, the successful response to the allocation is indicated on the Data Set Utility panel that reappears. You are returned to this after processing the Allocate New Data Set panel. The top right of the panel indicates Data set allocated.

We have now created a partitioned data set with the name, MIRIAM.PRIVATE.JCLLIB. This is an empty data set, so the first thing we do is add a member to this data set.

## 3.26 Edit function: Option 2

```
Menu Utilities Compilers Options Status Help
-----
                                ISPF Primary Option Menu
Option ==> 2_

0 Settings      Terminal and user parameters      User ID . . : MIRIAM
1 View          Display source data or listings      Time. . . . : 17:23
2 Edit          Create or change source data       Terminal. . : 3278
3 Utilities     Perform utility functions           Screen. . . : 1
4 Foreground   Interactive language processing     Language. . : ENGLISH
5 Batch         Submit job for language processing   Appl ID . . : PDF
6 Command      Enter TSO or Workstation commands   TSO logon : IKJACCT
7 Dialog Test  Perform dialog testing              TSO prefix: NEWPREF
8 LM Facility  Library administrator functions    System ID : SC43
9 IBM Products IBM program development products  MVS acct. : ACCNT#
10 SCLM        SW Configuration Library Manager   Release . . : ISPF 5.2
11 Workplace   ISPF Object/Action Workplace

Enter X to Terminate using log/list defaults

F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward  F9=Swap
F10=Actions  F12=Cancel
```

Figure 3-26 Edit: Option 2

You can add a new member to a PDS using ISPF Option 2 (Edit) in the ISPF Primary Option Menu. When you choose this option, the Entry Edit panel is displayed.

You can also use option 2 to change an existing member. This can be accomplished by entering the name of the data set and member you want to change in the Edit Entry Panel.

## 3.27 Edit Entry Panel

```
Menu  RefList  RefMode  Utilities  Workstation  Help
-----
Edit Entry Panel
Command ==> _____

ISPF Library:
Project . . . . MIRIAM
Group . . . . PRIVATE . . . . _____ . . . . _____
Type . . . . JCLLIB
Member . . . . abc1_____ (Blank or pattern for member selection list)

Other Partitioned, Sequential or VSAM Data Set:
Data Set Name . . . . _____
Volume Serial . . . . _____ (If not cataloged)

Workstation File:
File Name . . . . _____

Options
Initial Macro . . . . _____ - Confirm Cancel/Move/Replace
Profile Name . . . . _____ - Mixed Mode
Format Name . . . . _____ - Edit on Workstation
Data Set Password . . . . _____ - Preserve VB record length

F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward  F9=Swap
F10=Actions  F12=Cancel
```

Figure 3-27 Edit Entry Panel

In this panel you have to supply the name of the PDS to which you want to add a new member. In our example the PDS name is MIRIAM.PROJECT.JCLLIB (the data set we allocated previously).

Add a new member called ABC1. After entering the corresponding input fields in this panel, press Enter.

As discussed previously, if your data set has three qualifiers, you can use the Project/Group/Type fields to identify your data set. The advantage is that ISPF stores this information in your profile data set and the next time you enter this panel the fields are set by the saved information. If your data set does not have three qualifiers, you must use the Other Partitioned or Sequential Data Set field, embedding the data set name in single quotes, unless you want ISPF to add your prefix as the HLQ.



## 3.28 Editing a data set

```
File Edit Edit_Settings Menu Utilities Compilers Test Help
EDIT MIRIAM.PRIVATE.JCLLIB(ABC1) - 01.00 Columns 00001 00072
Command ==> Scroll ==> PAGE
***** ***** Top of Data *****
000100 PROC 0 DB
000200 IF &DB = .DB THEN +
.....
..... -
.....
.....
.....
***** ***** Bottom of Data *****

F1=Help      F2=Split    F3=Exit     F5=Rfind    F6=Rchange  F7=Up
F8=Down      F9=Swap     F10=Left    F11=Right   F12=Cancel
```

Figure 3-28 Editing a data set

After you press Enter a blank panel is displayed (Figure 3-28). Type in the records you want to create in the new member. In the example we created two records, 00100 and 00200.

You can use the Edit function to create, display, or change data stored in the partitioned data set member or sequential data sets. When the editor displays existing data, each line consists of a six-column Line Command field followed by a 72-column data field. To view data that is not displayed, use the scroll commands. The following are PDF default values for function keys:

- ▶ F7/19 Scrolls up
- ▶ F8/20 Scrolls down
- ▶ F10/22 Scrolls left
- ▶ F11/23 Scrolls right

To see the function key values, type **KEYS** in the Command line and press Enter.

You can issue line commands and primary commands in edit mode.

## 3.29 ISPF edit: Some line commands

Command	Description
I	Insert lines
D	Delete lines
R	Repeat lines
C	Copy lines
M	Move lines
A	After line
B	Before line
(	Shift right columns
<	Shift right data
)	Shift left columns
>	Shift left data
X	Exclude lines

Figure 3-29 ISPF - Edit: Some line commands

Line commands affect only a single line or block of lines. You enter line commands by typing them in the line command field and pressing Enter. With line commands you can:

- ▶ Insert or delete lines
- ▶ Repeat lines
- ▶ Rearrange lines or overlay portions of lines
- ▶ Simplify text entry and formatting
- ▶ Define an input mask
- ▶ Shift data
- ▶ Include or exclude lines from the display
- ▶ Control tabs and boundaries for editing
- ▶ Convert some types of special temporary lines to data lines

Figure 3-29 shows some line commands you can use. To learn about line commands:

1. Type ? in the line command field and press Enter. This causes a short Help message to appear at top right of the screen.
2. Press Help again and a long message appears.
3. Press Help again to display a help panel with all the line commands included.

There is also a help panel for each line command, showing its effect.

### 3.30 ISPF edit panel: Inserting lines

```
Screen 1

  File Edit Edit_Settings Menu Utilities Compilers Test Help
-----
ISREDDE2 MIRIAM.PRIVATE.JCLLIB(ABC1) - 01.03 Columns 00001 00072
Command ==> _____ Scroll ==> PAGE
***** ***** Top of Data *****
i50100 PROC 0 DB
000200 IF &DB = .DB THEN +
***** ***** Bottom of Data *****

Screen 2

  File Edit Edit_Settings Menu Utilities Compilers Test Help
-----
ISREDDE2 MIRIAM.PRIVATE.JCLLIB(ABC1) - 01.03 Columns 00001 00072
Command ==> _____ Scroll ==> PAGE
***** ***** Top of Data *****
000100 PROC 0 DB
.....
.....
.....
.....
.....
000200 IF &DB = .DB THEN +
***** ***** Bottom of Data *****
```

Figure 3-30 ISPF edit panel: Inserting lines

Figure 3-30 shows an example of the line command **Insert**. Type **I** to insert one line; to insert multiple lines type **Ixx**, where **xx** is the number of lines you want to insert.

We entered **I5** in the first line (screen 1) and five blank lines were added (screen 2).

### 3.31 ISPF edit: Repeating and deleting lines

```
Command ==> _____ Scroll ==> PAGE
***** ***** Top of Data *****
000100 PROC 0 DB
d00200 IF &DB = .DB THEN +
***** ***** Bottom of Data *****

Results in:

Command ==> _____ Scroll ==> PAGE
***** ***** Top of Data *****
000100 PROC 0 DB
***** ***** Bottom of Data *****

Issuing the R5 command to repeat the line 5 times.

Command ==> _____ Scroll ==> PAGE
***** ***** Top of Data *****
r50100 PROC 0 DB
***** ***** Bottom of Data *****

Results in:

Command ==> _____ Scroll ==> PAGE
***** ***** Top of Data *****
000100 PROC 0 DB
000200 PROC 0 DB
000300 PROC 0 DB
000400 PROC 0 DB
000500 PROC 0 DB
000600 PROC 0 DB
***** ***** Bottom of Data *****
```

Figure 3-31 ISPF Edit: Repeating and deleting lines

You can delete lines by issuing the line command **D** which will delete one line; to delete multiple lines, issue **Dxx**, where **xx** is the number of lines to delete. You can also delete a block of lines by using **DD** at the beginning and at the end of the block. In Figure 3-31 we deleted one line by entering **D** in the corresponding command line.

Repeating lines can be done by issuing **R** (one line), **Rxx** for **xx** lines, or **RR** for a block. Figure 3-31 shows the result of the execution of an **R5** command: the line is repeated five times.

## 3.32 Edit: Copying lines

```
Command ==> _____ Scroll ==> PAGE
***** ***** Top of Data *****
c00100 PROC 0 DB
000200 PROC 1 DB
a00300 PROC 2 DB
Results in:
Command ==> _____ Scroll ==> PAGE
***** ***** Top of Data *****
000100 PROC 0 DB
000200 PROC 1 DB
000300 PROC 2 DB
000310 PROC 0 DB
Issuing the CC command to copying a block
Command ==> _____ Scroll ==> PAGE
***** ***** Top of Data *****
cc0100 PROC 0 DB
cc0200 PROC 1 DB
b00300 PROC 2 DB
Results in:
Command ==> _____ Scroll ==> PAGE
***** ***** Top of Data *****
000100 PROC 0 DB
000200 PROC 1 DB
000210 PROC 0 DB
000220 PROC 1 DB
000300 PROC 2 DB
```

Figure 3-32 Edit - copying line

Copying lines can be done by issuing **C** to copy one line, **Cxx**, to copy xx lines, or **CC** to copy a block, with **CC** marking the first and last lines of the block to be copied. After indicating the lines, you must tell the Editor where to copy the lines to. This is done by typing either a **B** (for before) or an **A** (for after) on the line following or preceding where you want the data to be copied.

### 3.33 ISPF/PDF edit: Primary commands

```
ISR2M000 ----- Edit Primary Commands ----- Tutorial
Option ==> _____

The following topics are presented in sequence, or may be selected by number.
Individual commands may be selected by entering the command name.

  0 - Primary commands (general information)
  1 - Miscellaneous commands
  2 - FIND/CHANGE/EXCLUDE commands
  3 - Number control commands
  4 - Display mode control commands
  5 - Termination control commands
  6 - External data commands
  7 - Macro control commands
  8 - Data editing commands
  9 - Edit settings command

Note: Select >CANCEL to see information for the CANCEL command.
```

Figure 3-33 ISPF/PDF edit: Primary commands tutorial

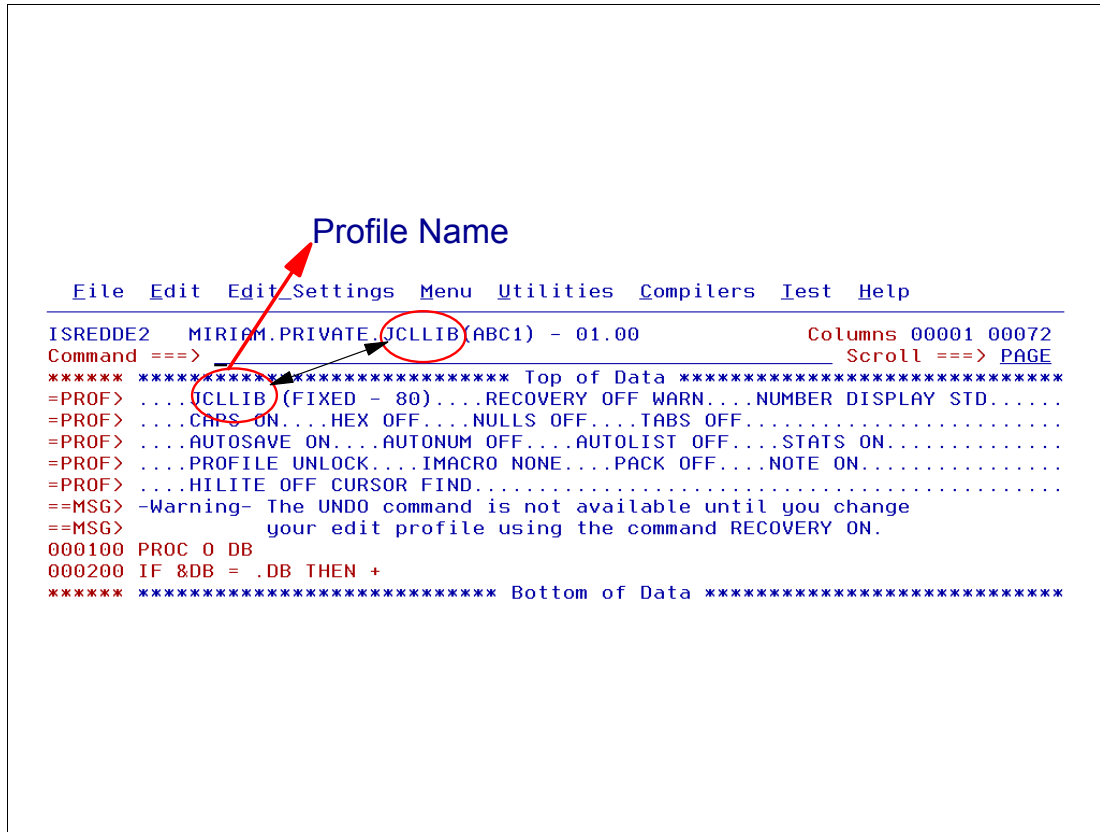
Primary commands affect the entire data set. You type them on the Command ==> field. You use a primary command to:

- ▶ Control your editing environment
- ▶ Find a specific line
- ▶ Find and change a character string
- ▶ Combine several members into one
- ▶ Split a member into two or more members
- ▶ Submit data to the job stream
- ▶ Save the edited data or cancel without saving
- ▶ Sort data
- ▶ Delete lines
- ▶ Access dialog element models
- ▶ Run an edit macro

An easy way to access a tutorial about all Edit primary commands is:

1. Type **P** in the Command line and press Enter. You receive the message COMMAND P NOT FOUND. Press Enter; you receive a short message at the right top of the panel.
2. Press the Help function key; now you receive a long message.
3. Press the Help function key one more time; now you are in the tutorial of all Edit primary commands showing how each command works.

### 3.34 ISPF/PDF edit: Profile command



The screenshot shows the ISPF/PDF edit profile command interface. At the top, a menu bar includes 'File', 'Edit', 'Edit Settings', 'Menu', 'Utilities', 'Compilers', 'Test', and 'Help'. Below the menu bar, the current session information is displayed: 'ISREDDE2 MIRIAM.PRIVATE.JCLLIB(ABC1) - 01.00' and 'Columns 00001 00072'. The 'Command ===>' field is empty, and the 'Scroll ===>' field is set to 'PAGE'. The main display area shows a list of profile options, with the first line highlighted: '\*\*\*\*\* Top of Data \*\*\*\*\*'. The options listed are: 'JCLLIB (FIXED - 80)...RECOVERY OFF WARN...NUMBER DISPLAY STD...', 'CAPS ON...HEX OFF...NULLS OFF...TABS OFF...', 'AUTOSAVE ON...AUTONUM OFF...AUTOLIST OFF...STATS ON...', 'PROFILE UNLOCK...IMACRO NONE...PACK OFF...NOTE ON...', and 'HILITE OFF CURSOR FIND...'. A warning message follows: '-Warning- The UNDO command is not available until you change your edit profile using the command RECOVERY ON.' Below the warning, the current profile settings are shown: '000100 PROC 0 DB' and '000200 IF 8DB = .DB THEN +'. The bottom of the display area is marked with '\*\*\*\*\* Bottom of Data \*\*\*\*\*'. Annotations include a red arrow pointing to the 'JCLLIB' field in the session information, and a red circle around the 'JCLLIB' field in the profile options list.

Figure 3-34 Profile edit primary command

Type the **PROFILE** primary command and press Enter. A set of highlighted lines shows the profile options you are using to edit the data set. Each Edit Profile has a name, which comes from the low level qualifier—JCLLIB in our example in Figure 3-34. To choose another existing profile or to change a profile name, type **PROF profname**.

The **RECOVERY ON** option permits you to recover the changes you made before a session failure. This option creates a copy of all changes you made after a **SAVE** command. When a session fails, the next time you return to the TSO/ISPF Edit function, you can edit the same data set with all updates and you can continue editing and save or cancel.

The **UNDO** command allows you to remove changes to a member during an edit session. Each **UNDO** issued removes the change performed since the previous Enter was done. Subsequent **UNDO** commands remove previous updates in sequence from the newest change to the oldest that has occurred during this edit session. **UNDO** can only be used with the **RECOVERY ON**, **SETUNDO REC**, or **SETUNDO STG** profile options. You can prepare profiles, give them names, and choose the appropriate profile to edit a data set by typing its name at the EDIT Entry Panel, PROFILE field.

To eliminate the highlighted lines, type **RESET** or **RES** in the Command line and press Enter.

### 3.35 ISPF/PDF edit: Saving new or updated files

- ❑ Saving your file
  - Enter **SAVE** on the command line
  - Use PF3 to save the file with profile option **AUTOSAVE ON**
- ❑ Canceling updates to a file
  - Enter **CANCEL** on the command line

Figure 3-35 ISPF/PDF: Save command

After updating:

- ▶ To save your data, issue the **SAVE** primary command. With the profile option **AUTOSAVE ON**, pressing the F3/F15 END key ends the edition and saves the data.
- ▶ Canceling your updates can be done by issuing the **CANCEL** primary command. This removes all changes made since the last **SAVE** was performed or since the data set was edited.



## 3.36 ISPF Data Set List Utility option

```
Menu RefList RefMode Utilities Help
-----
ISRUDLP                               Data Set List Utility
Option ==> _____

blank Display data set list           P Print data set list
  V Display VTOC information           PV Print VTOC information

Enter one or both of the parameters below:
Dsname Level . . . MIRIAM
Volume serial . . . _____

Data set list options
Initial View . . . 1
                   1. Volume           Enter "/" to select option
                   2. Space             / Confirm Data Set Delete
                   3. Attrib            / Confirm Member Delete
                   4. Total             / Include Additional Qualifiers

When the data set list is displayed, enter either:
"/" on the data set list command field for the command prompt pop-up,
an ISPF line command, the name of a TSO command, CLIST, or REXX exec, or
"=" to execute the previous command.

F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward  F9=Swap
F10=Actions  F12=Cancel
```

Figure 3-36 ISPF option 3.4 panel

The Data Set List Utility is a useful option in your daily job. From the Primary Option Menu select option 3.4 to use this list utility; the panel shown in Figure 3-36 is returned. You can use this option to manage all data sets that you have access to. Move the cursor to the Dsname Level field and enter the high level qualifier of the data sets you want to work with. First, take a look in the options available:

- ▶ You can obtain a list of all data sets in a DASD volume. To do this, use option **V** and enter a volume name.
- ▶ You can obtain a data set list.
- ▶ You can customize what kind of data set information you want.

For a data set name list, the data set name qualifiers can be partially specified using asterisks (\*) as global file-name characters and percent signs (%) as placeholders:

- \* A single asterisk indicates that at least one qualifier is needed to occupy that position. A single asterisk within a qualifier indicates that zero or more characters can occupy that position.
- \*\* A double asterisk indicates that zero or more qualifiers can occupy that position. A double asterisk within a qualifier is invalid.
- % A single percent sign indicates that any one character can occupy that position. One to eight percent signs can be specified in each qualifier.

For example, entering `ABC.DDD%*.E*%%F.**` lists all data sets with ABC as HLQ, second qualifier starts with DDD, the third starts with E and ends with F and has at least 3 characters between E and F. After that it can have any number of qualifiers. In Figure 3-36 we used the *MIRIAM* HLQ.

### 3.37 Working with a data set list

```

Menu Options View Utilities Compilers Help
-----
ISRUDSL0 Data Sets Matching MIRIAM                               Row 1 of 13
Command ==> _____ Scroll ==> PAGE

Command - Enter "/" to select action                               Message                               Volume
-----
MIRIAM                                                            *ALIAS
MIRIAM.BOOKS.E0Z2A100.PRINT                                       TOTSTG
MIRIAM.BROADCAST                                                  TOTTSU
MIRIAM.EOXMBMGR.PRINT                                             TOTST4
MIRIAM.HCD.MSGLOG                                                 TOTST1
MIRIAM.HCD.TERM                                                   TOTTSF
MIRIAM.HCD.TRACE                                                  TOTSTG
MIRIAM.OLD.DATASET.SAMPLE2                                        TOTTSF
E_ MIRIAM.PRIVATE.JCLLIB                                          TOTST2
MIRIAM.SC43.ISPF42.ISPPROF                                        TOTTS2
MIRIAM.SC52.ISPF42.ISPPROF                                        TOTTS6
MIRIAM.SC52.SPFTMP1.CNTL                                          TOTTS1
MIRIAM.SC62.ISPF42.ISPPROF                                        TOTTS7
***** End of Data Set List *****

F1=Help   F2=Split   F3=Exit   F5=Rfind   F7=Up     F8=Down   F9=Swap
F10=Left  F11=Right  F12=Cancel

```

Figure 3-37 Data set list for MIRIAM

Figure 3-37 shows all catalogued data sets having the high level qualifier MIRIAM.

You can select data sets for processing by entering any of the line commands shown in Figure 3-38 on page 127 in the left of the data set name. As you can see, this is a very comprehensive list of commands to enable you to manage your data sets. To learn about these commands, type **Help** in the Command field.

Typing **E** next to data set MIRIAM.PRIVATE.JCLLIB displays a list of the members contained within that data set. Once the member list is displayed, you can select a member by typing **S** in front of the member name. Also, you can perform many actions against the members, such as copy, rename, delete, and so forth. To see all actions available in the member list:

1. Type **?** at the left of any member name. You receive an error message.
2. Press the help function key; you receive a long error message.
3. Press the help function key again; the tutorial for the commands available is displayed. Note that in the upper right of the panel the following appears:

More:     +

It indicates a scrollable panel; you can go forward and backward throughout this panel using PF7 and PF8 (default keys).

### 3.38 Data Set List Actions

```

Menu  Options  View  Utilities  Compilers  Help
-
I  ISRUABC          Data Set List Actions
C
C  Data Set: MIRIAM.PRIVATE.JCLLIB
-
  DSLIST Action
  -
  1. Edit          12. Compress
  2. View          13. Free
  3. Browse        14. Print Index
  4. Member List   15. Reset
  5. Delete        16. Move
  6. Rename        17. Copy
  7. Info          18. Refadd
  8. Short Info    19. Exclude
  9. Print         20. Unexclude 'NX'
  10. Catalog      21. Unexclude first 'NXF'
  11. Uncatalog    22. Unexclude last 'NXL'

/
*  Select a choice and press ENTER to process data set action.
  F1=Help          F2=Split          F3=Exit           F7=Backward
  F8=Forward       F9=Swap           F12=Cancel
  -
  ow 1 of 13
  ==> PAGE

  Volume
  -----
  *ALIAS
  TOTSTG
  TOTTSU
  TOTST4
  TOTST1
  TOTTSA
  TOTSTG
  TOTTST
  TOTST2
  TOTTS2
  TOTTS7
  TOTTSI
  TOTTS7
  *****

```

Figure 3-38 Data Set List Actions

Typing a / on the MIRIAM.PRIVATE.JCLLIB line causes ISPF to display a panel with all the list actions you can use in that column. You can choose the option number or, by going back to the data set list, type the command in the Command column.

### 3.39 Job control language (JCL)

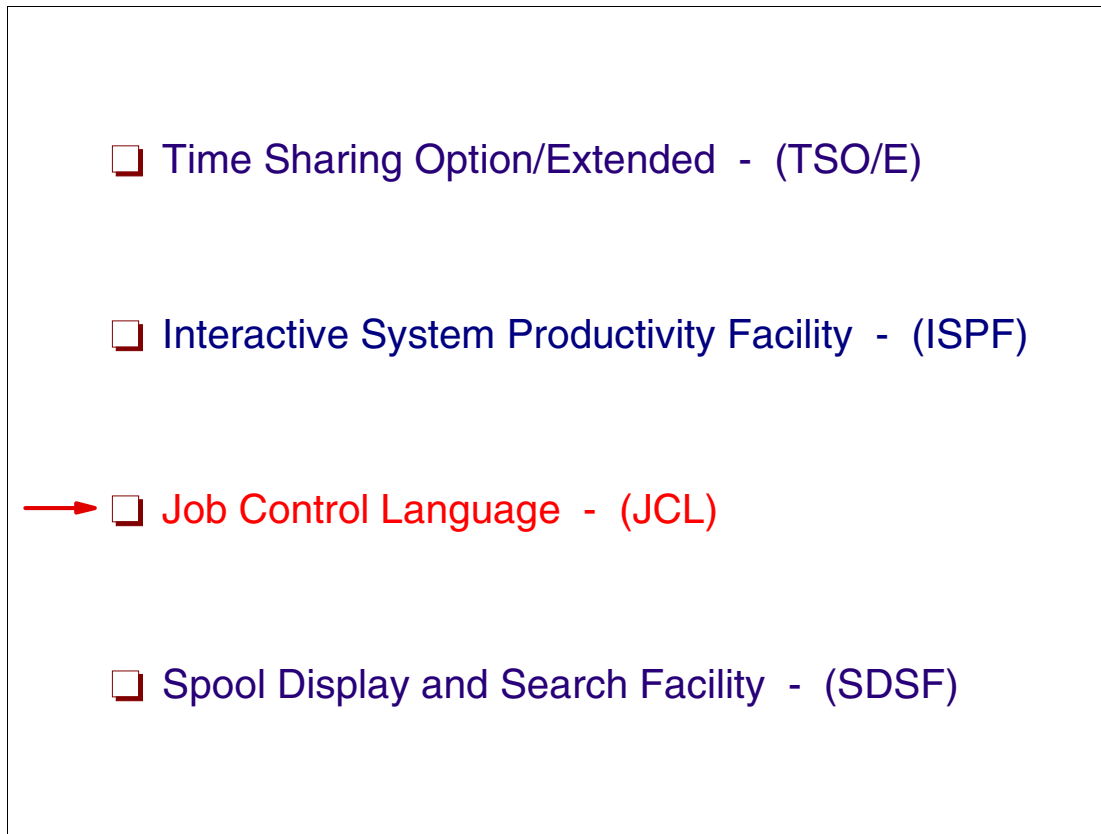


Figure 3-39 Job control language (JCL)

For your program to execute on the computer and perform the work you designed it to do, your program must be processed by your operating system.

Your operating system consists of an MVS/SP™ base control program (BCP) with a job entry subsystem (JES2 or JES3) and DFSMS/MVS DFSMSdfp installed with it.

For the operating system to process a program, programmers must perform certain job control tasks. These tasks are performed through the job control statements, which consist of:

- ▶ JCL statements
- ▶ JES2 control statements
- ▶ JES3 control statements

## 3.40 JCL introduction



Figure 3-40 Job Control Language - JCL

To get your MVS system to do work for you, you must describe to the system the work you want done and the resources your work needs. You use JCL to provide this information to MVS.

One way of thinking about JCL is to compare it to a menu in a restaurant. If you are a customer at a restaurant, you and the other customers don't just walk into the kitchen and start cooking your own dinners—that would defeat the very purpose of going to a restaurant. Instead, from a menu describing all the restaurant has to offer, you select items to make up an order, specifying which entrees you want, which salad dressing you prefer, and any other special requests you have. You then ask the waiter to take your order to the kitchen. In the kitchen, a team of chefs divides up the work and the appropriate ingredients in order to prepare each dish as quickly and efficiently as possible. While the meals are being prepared, you and your friends can ignore what's going on in the kitchen, engaging instead in dinner conversation, catching up on the latest news. When the waiter brings your meal out, you concentrate on your enjoyment of the meal.

Now imagine yourself back at the office using your MVS system, and think of JCL as the menu. In the same way that you and the other diners select items from the menu and place orders for the waiter to take to the team of chefs, you and other MVS users use JCL to define work requests (called jobs), and use a job entry subsystem (JES) to submit those jobs to MVS. Using the information that you and the other users provide with JCL statements, MVS allocates the resources needed to complete all of your jobs just as the kitchen chefs divided up the work to prepare the orders of all the customers.

And just as the chefs worked in the kitchen while you and the other diners devoted your attention to what was going on at your tables, MVS completes the submitted jobs in the background of the system, enabling you and the other users to continue working on other activities in the foreground.

And just as the waiter conveys the results of the chefs' work to you, JES presents the output of the jobs to you.

For a complete description of the process see *z/OS V1R2.0 MVS JCL User's Guide*, SA22-7598.

### 3.41 JCL-related actions

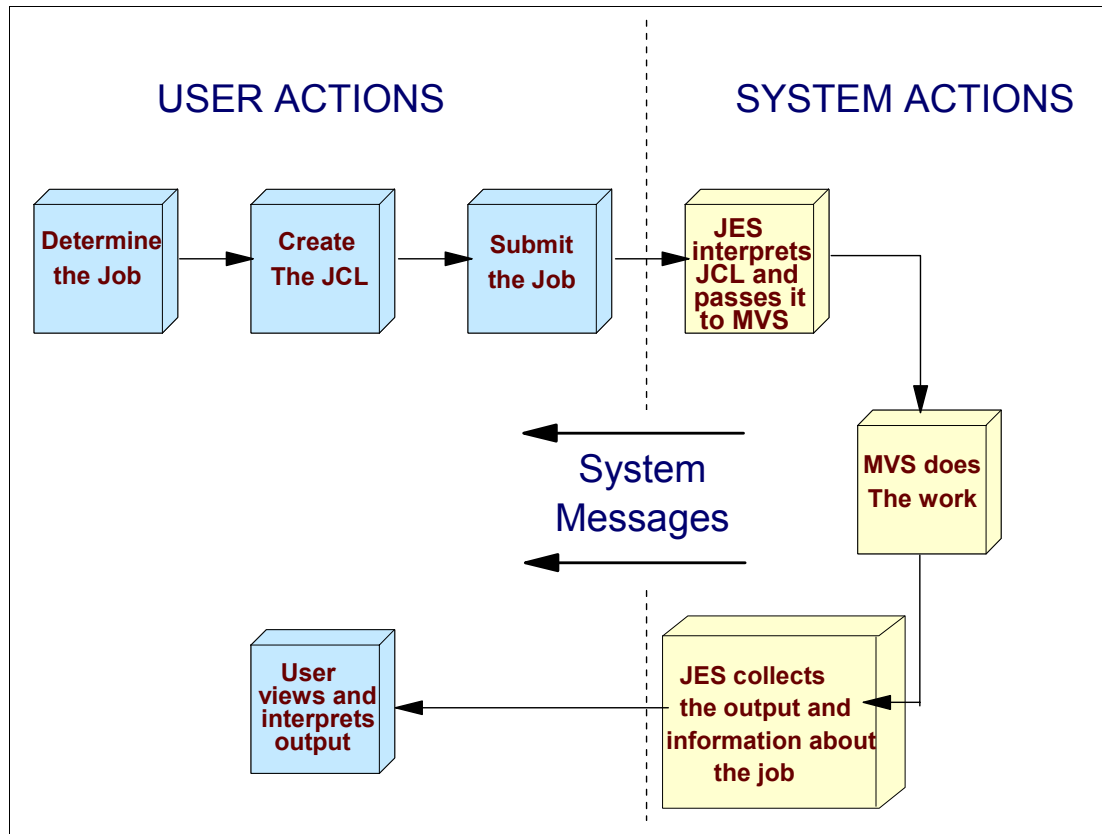


Figure 3-41 JCL-related actions

Figure 3-41 shows an overview of the job submission process. The user performs the activities on the left side of the figure, and the system performs those on the right. In this example, MVS and JES comprise the “system.”

For every job that you submit, you need to tell MVS where to find the appropriate input, how to process that input (that is, what program or programs to run), and what to do with the resulting output.

You use JCL to convey this information to MVS through a set of statements known as job control statements. JCL’s set of job control statements is quite large, enabling you to provide a great deal of information to MVS.

Most jobs, however, can be run using a very small subset of these control statements. Once you become familiar with the characteristics of the jobs you typically run, you may find that you need to know the details of only some of the control statements.

Within each job, the control statements are grouped into job steps. A job step consists of all the control statements needed to run one process, for example a sort, a copy, or an application program. If a job needs to run more than one process, the job will contain a different job step for each of those programs. A job can have from one up to 255 steps.

## 3.42 Required control statements

- Control statements
  - Job (JOB)
  - Execute (EXEC)
  - Data Definition (DD)

*Figure 3-42 Required control statements*

Every job must contain, at minimum, the following two types of control statements:

- ▶ A JOB statement, to mark the beginning of a job and assign a name to the job. The JOB statement is also used to provide certain administrative information, including security, accounting, and identification information. Every job has one and only one JOB statement.
- ▶ An EXEC (execute) statement, to mark the beginning of a job step, to assign a name to the step, and to identify the program or procedure to be executed in the step. You can add various parameters to the EXEC statement to customize the way the program executes. Every job has at least one EXEC statement.

In addition to the JOB and EXEC statements, most jobs usually also contain:

- ▶ One or more DD (data definition) statements, to identify and describe the input and output data to be used in the step. The DD statement may be used to request a previously-created data set, to define a new data set, to define a temporary data set, or to define and specify the characteristics of the output.



### 3.43 JCL streams and jobs

- ❑ Programmers create jobs
  - Using JCL statements
- ❑ Jobs are submitted to and processed by:
  - JES2 or JES3
  - z/OS operating system
- ❑ Jobs have one or more job steps

Figure 3-43 JCL streams and jobs

For the operating system to process a program, system programmers or application programmers must perform certain job control tasks. These tasks are performed through the job control statements, which consist of:

1. JCL statements, as mentioned before
2. JES2 control statements *or* JES3 control statements

MVS uses a job entry subsystem (JES) to receive jobs into the operating system, schedule them for processing by MVS, and control their output processing. JES2 is descended from HASP (Houston automatic spooling priority). HASP is defined as a computer program that provides supplementary job management, data management, and task management functions such as: scheduling, control of job flow, and spooling. HASP remains within JES2 as the prefix of most module names and the prefix of all messages sent by JES2 to the operator.

JES2 (job entry subsystem 2) is a functional extension of the HASP II program that receives jobs into the system and processes all output data produced by the job. So, what does all that mean? Simply stated, JES2 is that component of MVS that provides the necessary functions to get jobs into, and output out of, the MVS system. It is designed to provide efficient spooling, scheduling, and management facilities for the MVS operating system.

But, none of this explains why MVS needs a JES. Basically, by separating job processing into a number of tasks, MVS operates more efficiently. At any point in time, the computer system resources are busy processing the tasks for individual jobs, while other tasks are waiting for

those resources to become available. In its most simple view, MVS divides the management of jobs and resources between the JES and the base control program of MVS. In this manner, JES2 manages jobs before and after running the program; the base control program manages them during processing.

### **JES2 compared to JES3**

IBM provides two JESs from which to choose: JES2 and JES3. In an installation that has only one processor (computer), JES2 and JES3 perform similar functions. That is, they read jobs into the system, convert them to internal machine-readable form, select them for processing, process their output, and purge them from the system. However, for an installation that has more than one processor in a configuration, there are noticeable differences in how JES2 exercises independent control over its job processing functions. That is, within the configuration, each JES2 processor controls its own job input, job scheduling, and job output processing. In a sysplex environment, it is possible to configure JES2 to share spool and checkpoint data sets with other JES2 systems in the same sysplex. This configuration is called Multi-Access Spool (MAS).

In contrast, JES3 exercises centralized control over its processing functions through a single global JES3 processor. This global processor provides all job selection, scheduling, and device allocation functions for all the other JES3 systems. The centralized control that JES3 exercises provides increased job scheduling control, deadline scheduling capabilities, and increased control by providing its own device allocation.

## 3.44 JES control statements in JCL

### □ Controls the input and output processing of jobs

#### ➤ JES2

- /\*JOBPARM
- /\*MESSAGE
- /\*ROUTE
- /\*XEQ
- etc...

#### ➤ JES3

- /\*OPERATOR
- /\*ROUTE XEQ
- /\*PROCESS
- /\*MAIN
- etc ...

Figure 3-44 Coding JES2 and JES3 control statements in a JCL

Code JES control statements after JOB card JCL statement. The exception is for /\*PAUSE, JES3 appears before the JOB statement or between its continuation. JES2 and JES3 ignore control statements in a procedure.

The rules for coding JES2 control statements are the same as the rules for JCL statements, with the following additions:

- ▶ Columns 1 and 2 always contain the characters /\*
- ▶ Where comments are needed, code a JCL comment statement: /\*.
- ▶ If you code the same parameter on the same statement more than once, JES2 uses the value in the last parameter.

The rules for coding JES3 control statements are the same as the rules for JCL statements, with the following additions:

- ▶ Generally, the characters /\* are in columns 1 to 3. Some JES3 control statements *must* contain only a single /\* in columns 1 and 2.
- ▶ Columns 3 and 4 must not be blank.
- ▶ To code a comment on a JES3 control statement, code a blank after the control statement, and end the comment before column 72.

## 3.45 Introduction to JCL: Creating a data set

- Coding rules
- Creating a data set using JCL

```
EDIT          MIRIAM.PRIVATE.JCLLIB(JOB1) - 01.05          Columns 00001 00072
Command ==> _____ Scroll ==> HALF
***** ***** Top of Data *****
000001 //MIRIAM2 JOB 19,MIRIAM,NOTIFY=&SYSUID,MSGCLASS=T,
000002 // MSGLEVEL=(1,1),CLASS=A
000003 //STEP1 EXEC PGM=IEFBR14
000004 //*-----*
000005 //* THIS IS AN EXAMPLE OF A NEW DATA SET ALLOCATION
000006 //*-----*
000007 //NEWDD DD      DSN=MIRIAM.IEFBR14.TEST.NEWDD,
000008 //              DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,
000009 //              SPACE=(CYL,(10,10,45)),LRECL=80,BLKSIZE=3120
***** ***** Bottom of Data *****
```

Figure 3-45 Creating jobs: An introduction to JCL

First of all, you need to know the basic JCL codification rules. The JCL statements:

- ▶ Start in column 1, and are identified by a // at the beginning of the line
- ▶ The commentary lines are identified by a /\* at the start of the line
- ▶ Are coded from column 1 to column 71
- ▶ A comma indicates that the statement has continuation
- ▶ A continuation of a statement *must* start between columns 4 and 16
- ▶ // and the rest of statement with blanks indicates the end of the job
- ▶ Key words are in uppercase

Comments must be separated from operators parameter by at least one blank.

In “Allocating data sets: Utility option” on page 109, you learned how to allocate a data set using ISPF dialogs. Now, we show you how you can create a data set using JCL control statements.

Besides submitting jobs to a system, you can use JCL statements to create, delete, catalog, or uncatalog data sets. Figure 3-45 shows the JCL we use to create a data set and to review some of the fundamental JCL statements. For more details refer to *z/OS V1R4.0 MVS JCL Reference, SA22-7597*.

## 3.46 JCL: JOB statement

- ❑ Create a member using ISPF edit
- ❑ Create JCL statements
  - JOB card statement ←
  - EXEC statement
  - DD statement

```
EDIT          MIRIAM.PRIVATE.JCLLIB(JOB1) - 01.05          Columns 00001 00072
Command ==> _____ Scroll ==> HALF
***** Top of Data *****
000001 //MIRIAM2 JOB 19,MIRIAM,NOTIFY=&SYSUID,MSGCLASS=T,
000002 // MSGLEVEL=(1,1),CLASS=A
000003 //STEP1 EXEC PGM=IEFBR14
000004 //*-----*
000005 //* THIS IS AN EXAMPLE OF A NEW DATA SET ALLOCATION
000006 //*-----*
000007 //NEWDD DD      DSN=MIRIAM.IEFBR14.TEST.NEWDD,
000008 //              DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,
000009 //              SPACE=(CYL,(10,10,45)),LRECL=80,BLKSIZE=3120
***** Bottom of Data *****
```

Figure 3-46 The JOB card

Create a member to insert your JCL control statements. You already know how to create a member in a partitioned data set (see “Edit Entry Panel” on page 116 to review). The first statement you code is the JOB statement, also known as the JOB card. The JOB statement marks the beginning of a job and tells the system how to process the job.

In our exercise of creating a job using JCL, we used the JCL shown on the box in Figure 3-46. The JOB card and its parameters are highlighted by the inner box (lines 000001 and 000002).

The first line of the JOB1 member is the job statement. It specifies parameters to be used by the job entry subsystem to schedule this job for processing. The format of the job card, and the importance of the data specified in the job card vary from installation to installation. The following fields are important:

**Jobname** The first field is the jobname, in this case MIRIAM2. It can have up to eight characters. Some sites perform security checking against the job name to ensure standards, usually the ID of the user who submitted the job. Let us suppose the standard is: user ID suffixed with at least one alphanumeric character. If MIRIAM is the user ID, MIRIAM2 matches the standards. Other sites might not have any job name restrictions.

**JOB** Identifies the job to the system, when submitted. It must be present, must follow the jobname and there must be at least one space between them.

<b>Account</b>	Some sites use this field for accounting and job processing information. In the example, the value is 19.
<b>Programmer name</b>	The next field is the programmer name field, which you can use to identify the member name, for example. In the example, MIRIAM is the member name.
<b>Notify</b>	Tells the system where to send “job complete” information. &SYSUID tells the system to automatically insert your user ID here, so the information will be sent to you.
<b>Msgclass class</b>	MSGCLASS= assigns the job log to an output class. The output class and its characteristics are identified in a parameter file used at JES initialization.
<b>Msglevel</b>	Tells the system to reproduce this JCL code in the output, and to include allocation messages.
<b>Job class</b>	The CLASS= field identifies the JES job class this job will execute under. In the example, CLASS=A is the JES job class. Many sites do not use this option, and the JES class is set according to your user ID. Job classes are set up at JES initialization.

## 3.47 JCL: EXEC statement

- ❑ EXEC statement ←
- Region size
  - REGION=4096K
- Process conditions from previous steps
  - COND=(0,NE)
  - COND=(0,NE,STEP1)

```
EDIT          MIRIAM.PRIVATE.JCLLIB(JOB1) - 01.05          Columns 00001 00072
Command ==> _____ Scroll ==> HALF
***** ***** Top of Data *****
000001 //MIRIAM2 JOB 19,MIRIAM,NOTIFY=&SYSUID,MSGCLASS=T,
000002 // MSGLEVEL=(1,1),CLASS=A
000003 //STEP1 EXEC PGM=IEFBR14
000004 //*-----*
000005 //* THIS IS AN EXAMPLE OF A NEW DATA SET ALLOCATION
000006 //*-----*
000007 //NEWDD DD      DSN=MIRIAM.IEFBR14.TEST.NEWDD,
000008 //              DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,
000009 //              SPACE=(CYL,(10,10,45)),LRECL=80,BLKSIZE=3120
***** ***** Bottom of Data *****
```

Figure 3-47 EXEC statement parameters

The EXEC Statement identifies the step and the program to be executed. In our job, we have just one step, arbitrarily identified by the name STEP1, and we invoke the IEFBR14 program. This program comes in all installations and it does nothing, just receives the control and gives it back to the system; that's why it is good for using some JCL capabilities.

In the EXEC you specify:

- Step name** In the example STEP1 gives a name to the step. You could have called the step name, IEFBR14, or any name that will help you identify the step. In a large job, with many steps, unique step names can assist you when diagnosing problems. The choice is up to you.
- EXEC** Identifies a step job. It must be present.
- PGM=** Specifies the name of the program to be executed. In this case the program name is IEFBR14.

Two other parameters that you might use on the EXEC are:

- ▶ The *REGION* parameter specifies the quantity of virtual storage (or central storage when ADDRSPC=REAL is coded) a step requires. If no REGION parameter is specified, the system uses an installation default specified at JES initialization. Some programs may need more storage than is allowed by default. To permit the program to get more storage, you can code the REGION parameter as follows:

```
//STEP1 EXEC PGM=progrname,REGION=4096K.
```

This permits the programs to get up to 4 MB of storage below 16 MB and up to the installation default storage above 16 MB (IBM default is 32 MB).

- ▶ The *COND* parameter is used to inform the system to test return codes from previous job steps and determine whether to bypass this job step. You can specify one or more tests on the *COND* parameter, and you can test return codes from particular job steps or from every job step that has completed processing. If the test conditions are satisfied, the system evaluates the *COND* parameter as *true* and *bypasses* the job step. If the test conditions are not satisfied, the system evaluates the *COND* parameter as *false* and executes the job step. For example:

```
//STEP2 EXEC PGM=IEFBR14,COND=(0,NE)
```

With this statement, the system checks the return code from *all* previous steps, and if they were *not equal* to zero, then does not execute this step. You could also check for return codes that are *Equal* to, *GT* (greater than), *LT* (less than), *GE* (greater than or equal to), and *LE* (less than or equal to). You can check the return code in a specific step by coding:

```
//STEP2 EXEC PGM=IEFBR14,COND=(0,NE,STEP1)
```

With this statement, if STEP1 ends with return code zero, then STEP2 is executed.



## 3.48 JCL: DD statement

- ❑ DD statement ←
- DD name
  - As referred by the program
- ❑ DSN=
  - The data set name

```
EDIT          MIRIAM.PRIVATE.JCLLIB(JOB1) - 01.05          Columns 00001 00072
Command ==> _____ Scroll ==> HALF
***** ***** Top of Data *****
000001 //MIRIAM2 JOB 19,MIRIAM,NOTIFY=&SYSUID,MSGCLASS=T,
000002 // MSGLEVEL=(1,1),CLASS=A
000003 //STEP1 EXEC PGM=IEFBR14
000004 //*-----*
000005 //* THIS IS AN EXAMPLE OF A NEW DATA SET ALLOCATION
000006 //*-----*
000007 //NEWDD DD      DSN=MIRIAM.IEFBR14.TEST.NEWDD,
000008 //              DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,
000009 //              SPACE=(CYL,(10,10,45)),LRECL=80,BLKSIZE=3120
***** ***** Bottom of Data *****
```

Figure 3-48 JCL: DD statement parameters

It should be reiterated at this time that the program IEFBR14 actually does nothing. But it enables you to submit a valid job and the system can process the DD statements identified in the JCL. This allows you to allocate, delete, catalog, and uncatalog data sets by using a batch process. At job initialization, once the program IEFBR14 has been located, the system allocates the data sets specified in the DD statements, then the control is passed to program IEFBR14, which does nothing, and you get notified that your job is done.

The DD (data definition) statement describes a data set and specifies the input and output resources needed for the data set. The DD statement is highlighted in Figure 3-48 and identifies the data set we want to create.

This statement shows some of the parameters that can be coded in a DD statement.

**DDname** Used to give a name to DD. NEWDD is the name assigned in the example. The programs refer to DD names instead of dsnames. So, unless a program allocates dynamically, all ddnames referred by a program must be coded. For example, if a program in a step needs a file identified as OUTFILE, you must code a DD named OUTFILE, identifying the relevant data set. In the example, IEFBR14 does not use data sets, so you can choose any ddname you want.

**DSN** Used to identify the data set. In the example MIRIAM.IEFBR14.TEST.NEWDD is the data set.

## 3.49 DD statement parameters: DISP, UNIT

- ❑ DISP parameter
  - Control concurrent access to data set
  - What to do with the data set at end of processing
- ❑ UNIT parameter
  - What type of device is the data set allocated on

```
EDIT      MIRIAM.PRIVATE.JCLLIB(JOB1) - 01.05          Columns 00001 00072
Command ==> _____ Scroll ==> HALF
***** ***** Top of Data *****
000001 //MIRIAM2 JOB 19,MIRIAM,NOTIFY=&SYSUID,MSGCLASS=T,
000002 // MSGLEVEL=(1,1),CLASS=A
000003 //STEP1 EXEC PGM=IEFBR14
000004 //*-----*
000005 //* THIS IS AN EXAMPLE OF A NEW DATA SET ALLOCATION
000006 //*-----*
000007 //NEWDD DD   DSN=MIRIAM.IEFBR14.TEST.NEWDD,
000008 //           DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,
000009 //           SPACE=(CYL,(10,10,45)),LRECL=80,BLKSIZE=3120
***** ***** Bottom of Data *****
```

Figure 3-49 JCL DD statement parameters: DISP, UNIT

### DD statement: DISP parameter

The DISP parameter describes the status of a data set to the system and tells what to do with the data set after termination of the step or job. You specify this value for both normal and abnormal termination.

The first field identifies the *STATUS* of the data set and how to control access to it. It can be:

- NEW** Indicates the data set will be created and the job will have exclusive control of the data set. That means no other job can access this data set until the last step in this job that refers to this data set ends. NEW is the default.
- OLD** Indicates the data set exists and the job requires exclusive access to it.
- MOD** Indicates that if the data set exists, data will be appended to the end of the data set; otherwise, a new data set will be created. The job requires exclusive access to the data set.
- SHR** Indicates that the data set can be shared by other users.

The second field in the DISP parameter indicates to the system what to do with the data set when the step finishes *NORMAL*. It can be:

- CATLG** Catalog the data set
- UNCATLG** Uncatalog the data set
- DELETE** Delete the data set

**PASS**        Pass the data set to the subsequent steps  
**KEEP**        Keep the data set intact

The third field in the DISP parameter indicates the *ABNORMAL* completion action. It can be: DELETE, CATLG, UNCATLG, or KEEP.

In the example, the status field specifies to create the data set; in the normal termination of the step, to catalog; and in abnormal termination to delete the data set. To delete a data set that exists you code its DSN and DISP=(OLD,DELETE,DELETE).

### **DD statement: UNIT parameter**

This parameter identifies the device or type of device on which the data set will be allocated. You use this parameter to specify the number of devices to be used. If the data set exists, you only need to specify the device type if the data is not cataloged. Most installations now administer disk storage with the Storage Management System (DFSMS/MVS). With SMS, you do not need to use the UNIT parameter to specify a device for SMS-controlled data sets. Some common examples are:

**UNIT=SYSDA**        Allocates the data set on a direct access device (DASD)  
**UNIT=3390**         Allocates the data set on a 3390 type disk  
**UNIT=SYSALLDA**    Allocates the data set on a direct access device (DASD)  
**UNIT=TAPE**        Allocates the file on a TAPE device

## 3.50 DD statement parameters: SPACE, LRECL, BLKSIZE

- ❑ SPACE parameter
  - How much space to give the data set
- ❑ LRECL parameter
  - What is the size of each record in the data set
- ❑ BLKSIZE parameter
  - What is the block size (records in a block)

```
EDIT          MIRIAM.PRIVATE.JCLLIB(JOB1) - 01.05          Columns 00001 00072
Command ==> _____ Scroll ==> HALF
***** ***** Top of Data *****
000001 //MIRIAM2 JOB 19,MIRIAM,NOTIFY=&SYSUID,MSGCLASS=T,
000002 // MSGLEVEL=(1,1),CLASS=A
000003 //STEP1 EXEC PGM=IEFBR14
000004 //*-----*
000005 //* THIS IS AN EXAMPLE OF A NEW DATA SET ALLOCATION
000006 //*-----*
000007 //NEWDD DD      DSN=MIRIAM.IEFBR14.TEST.NEWDD,
000008 //              DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,
000009 //              SPACE=(CYL,(10,10,45)),LRECL=80,BLKSIZE=3120
***** ***** Bottom of Data *****
```

Figure 3-50 DD statement parameters: SPACE, LRECL, BLKSIZE

### DD statement: SPACE parameter

The *SPACE* DD parameter is required for allocating data sets on DASD. It identifies the space allocation required for your data set. Before a data set can be created on disk, the system must know how much space the data set will require and how the space is to be measured. You can code it as follows:

```
SPACE=(type,(primary-qty,second-qty,directory))
```

Where *type* can be:

- |                      |   |
|----------------------|---|
| <b>TRK</b>           | Requests that space be allocated in tracks.   |
| <b>CYL</b>           | Requests that space be allocated in cylinders.  |
| <b>block length</b>  | Specify here the block length to request an allocation in number of blocks. Indicates that the values specified for primary and secondary allocations are block quantities, and directs the system to compute the number of tracks to allocate using a block length. For example, <code>SPACE=(3150,(5,1))</code> means that the system will allocate five blocks, each block having 3150 bytes, as primary space and one block as secondary space. |
| <b>record length</b> | Specifies that the average record length in bytes will be used to allocate space. This is only applicable if SMS is active and the <code>AVGREC</code> parameter is coded.  |

The system allocates DASD space in whole tracks. The number of tracks required depends on how the records are blocked.

- primary-qty** Specifies the initial allocation amount.
- secondary-qty** Specifies an additional allocation amount. The system does not allocate additional space until it is needed.
- directory** You must code for a partitioned data set, to indicate the number of blocks the system must reserve for the directory. Partitioned Data Sets Extended (PDSE) grow dynamically; if you specify directory size, SMS uses the size you specify only if you later convert the PDSE to a PDS. Omit this parameter for sequential data sets.

Some examples of how you can code are:

- SPACE=(TRK,4) To allocate space to a sequential data set, requesting four tracks, only primary space.
- SPACE=(CYL,(5,2)) To allocate space to a sequential data set, five cylinders as primary allocation space and two cylinders as secondary.
- SPACE=(CYL,(10,,100)) To allocate space to a partitioned data set, only primary allocation of 100 cylinders and 100 directory blocks.

### **LRECL and BLKSIZE parameters**

Programs access data sets through ddnames. The ddnames are defined in the programs. Like them, data set characteristics like data set organization, logical record size and record size are also defined for the programs. During the allocation process you have to specify some of these characteristics. Some DD statement parameters you can use to specify the data set characteristics are:

- LRECL** Identifies the data set logical record size.
- BLKSIZE** Specifies the maximum length, in bytes, of a block.
- RECFM** Identifies the record format.
- DSORG** Identifies the data set organization. If you do not specify DSORG, the system uses the information in SPACE to determine if the data set should be sequential or partitioned

These parameters are part of Data set Control Block (DCB) information and can be coded in the JCL as follows:

```
// DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB,DSORG=PO)
```

or as individual parameters, without the need to specify the DCB=(...) parameter, for example:

```
// LRECL=80,  
// BLKSIZE=3120,  
// RECFM=FB,  
// DSORG=PO
```

If BLKSIZE is not specified, the system will determine what it considers to be an optimum block size for the device type on which the data set will be allocated. For a data set with fixed record format (all records with same size), the BLKSIZE must be a multiple of LRECL. For variable record size, the BLKSIZE must be a multiple of the greatest record plus four.

## 3.51 Submitting a job

```
EDIT          MIRIAM.PRIVATE.JCLLIB(JOB1) - 01.06          Columns 00001 00072
Command ==> submit          Scroll ==> HALF
***** ***** Top of Data *****
000001 //MIRIAM2 JOB 19,MIRIAM,NOTIFY=&SYSUID,MSGCLASS=T,
000002 // MSGLEVEL=(1,1),CLASS=A
000003 //STEP1 EXEC PGM=IEFBR14
000004 //*-----*
000005 //* THIS IS AN EXAMPLE OF A NEW DATA SET ALLOCATION
000006 //*-----*
000007 //NEWDD DD    DSN=MIRIAM.IEFBR14.TEST.NEWDD,
000008 //          DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,
000009 //          SPACE=(CYL,(10,10,45)),LRECL=80,BLKSIZE=3120
000010 //SYSPRINT DD  SYSOUT=A
000011 /*
***** ***** Bottom of Data *****

IKJ56250I JOB MIRIAM2(JOB26044) SUBMITTED
*** _
```

Figure 3-51 Submitting a job

You have now created the data set MIRIAM.PRIVATE.JCLLIB, built your JCL member JOB1, and now you are ready to SUBMIT this job.

To submit the JCL stream, enter **submit (sub)** on the command line. The TSO/E command processor sends the JCL statements to JES.

After entering the command, you receive the following message indicating that your job was submitted successfully, as shown in Figure 3-51:

```
JOB jobname (jobnumber) SUBMITTED
```

## 3.52 Spool Display and Search Facility (SDSF)

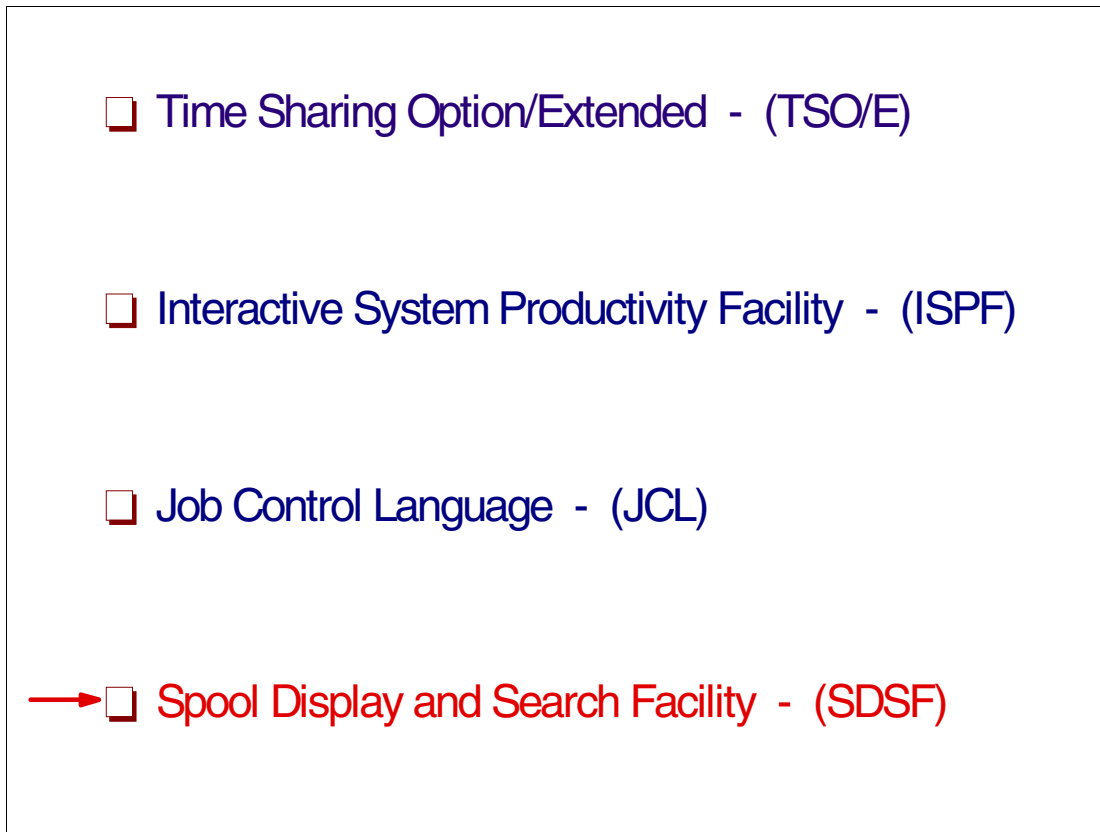


Figure 3-52 Spool Display and Search Facility (SDSF)

SDSF is a licensed program that helps authorized users efficiently monitor and control the operation of an MVS/JES2 system.

With SDSF you can authorize your MVS/JES2 users to:

- ▶ Control job processing (hold, release, cancel, and purge jobs)
- ▶ Monitor jobs while they are being processed
- ▶ Display job output before deciding to print it
- ▶ Manage the system's workflow
- ▶ Control the order in which jobs are processed
- ▶ Determine the number of output jobs and the total number of records to be printed
- ▶ Control the order in which output is printed
- ▶ Control printers and initiators
- ▶ View the MVS\* system log online and use commands to search for specific information in the log
- ▶ Dynamically change job data set output descriptors
- ▶ Issue JES2 and MVS commands that affect their jobs
- ▶ Print selected lines of the JES2 output data set
- ▶ Edit JCL direct from spool

### 3.53 SDSF: Panels hierarchy

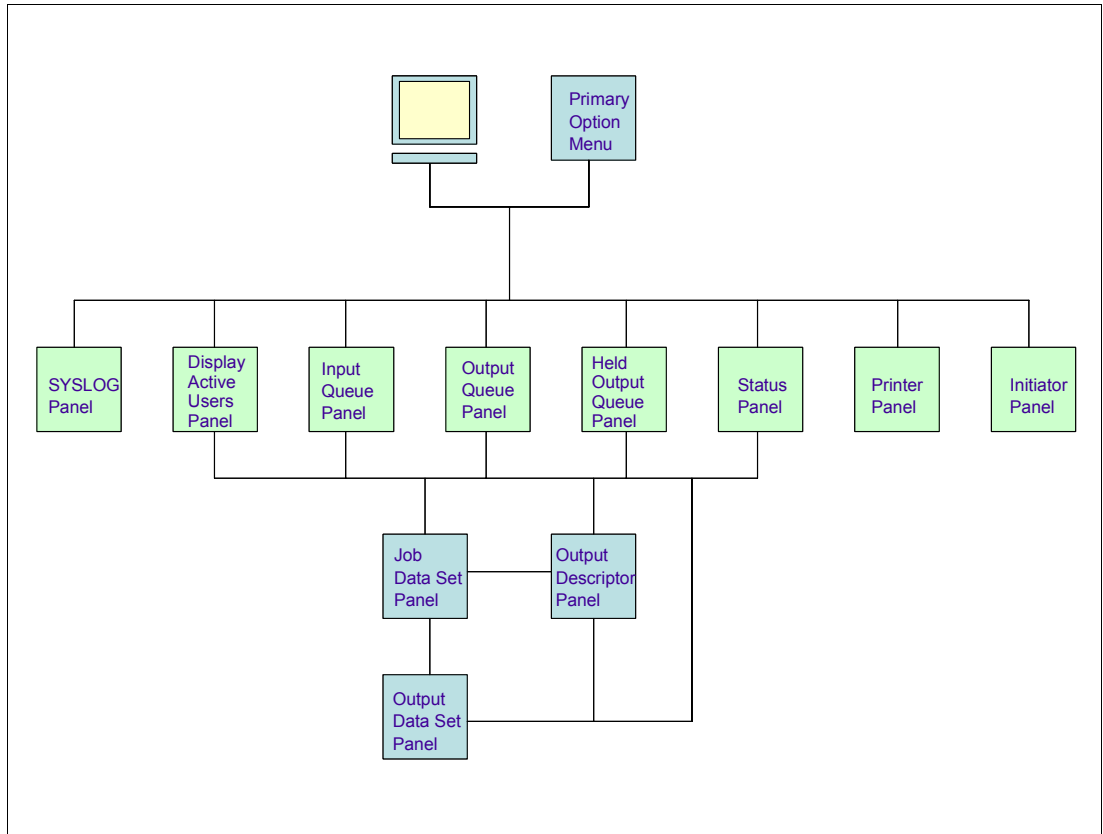


Figure 3-53 SDSF: Panels hierarchy

SDSF consists of panels that provide immediate information about jobs, printers, queues, and resources in an MVS/JES2 system. The SDSF panel hierarchy is illustrated in Figure 3-53. From these panels, authorized users can enter SDSF commands to control the processing of jobs and the operation of system resources. Authorized users also can issue MVS and JES2 system commands from the SDSF panels.



## 3.54 SDSF: Primary option menu

```
      Display Filter View Print Options Help
-----
ISFPCU41 ----- SDSF PRIMARY OPTION MENU -----
COMMAND INPUT ==> _                               SCROLL ==> PAGE

DA   Active users                               INIT  Initiators
I    Input queue                                PR    Printers
O    Output queue                               PUN   Punches
H    Held output queue                         RDR   Readers
ST   Status of jobs                            LINE  Lines
                                           NODE  Nodes
LOG   System log                               SO    Spool offload
SR   System requests                          SP    Spool volumes
MAS   Members in the MAS
JC   Job classes                               ULOG  User session log
SE   Scheduling environments
RES  WLM resources
ENC  Enclaves
PS   Processes

END   Exit SDSF

Licensed Materials - Property of IBM

5694-A01 (C) Copyright IBM Corp. 1981, 2002. All rights reserved.
US Government Users Restricted Rights - Use, duplication or
disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

F1=HELP      F2=SPLIT      F3=END      F4=RETURN   F5=IFIND    F6=B00K
F7=UP        F8=DOWN       F9=SWAP    F10=LEFT   F11=RIGHT   F12=RETRIEVE
```

Figure 3-54 SDSF: Primary option menu

In this section we discuss the SDSF facility and how you can use its functions to monitor and manage your workloads. We see how SDSF can be used to display job input and output data, and purge (delete) jobs that are on the input, output, or held queues. We review the monitoring functions of SDSF that enable you to evaluate the current workload, and enable you to cancel, hold, or reschedule work.

SDSF can be invoked from ISPF menus, but the setting of the options is often customized by each site differently. You will have to review your site's ISPF menus to find the SDSF option. Alternatively, issuing the **TSO SDSF** command, from the Command ==> line invokes SDSF. After choosing this option, the panel you receive will be similar to the one in Figure 3-54. However, it may not have all the same options shown in the figure; the options vary according to the security level of the user. The authority to perform functions within these options also varies according to the security level of the user. It is possible to control most system functions by using the SDSF facility. The scope of the functions includes reviewing job output, controlling the processing of jobs (both their input and output), printer control, operator functions, and system administration.

## 3.55 SDSF: Options menu

```
Display Filter View Print Options Help
-----
HQX7707 ----- SD
COMMAND INPUT ==>

DA   Active users
I    Input queue
O    Output queue
H    Held output queue
ST   Status of jobs

LOG  System log
SR   System requests
MAS  Members in the MAS
JC   Job classes
SE   Scheduling environments
RES  WLM resources
ENC  Enclaves
PS   Processes

END  Exit SDSF

1. Set action character display...
2. Find limit...
3. Change include SYSIN to ON
4. Set bookshelf...
5. Set display values to OFF
6. Set screen characteristics...
7. Set delay for responses...
8. Set communications timeout...
9. Set console name...
10. Set search characters...
11. Assign PF keys...
12. Change show PF keys to OFF
13. Set language for help and tutorial...
14. Set cursor to OFF
15. Set confirmation to OFF
16. Operlog limit for filter...
17. Set date format...
18. Set log default...

F1=HELP   F2=SPLIT   F3=END     F4=RETURN   F5=IFIND   F6=B00K
F7=UP     F8=DOWN    F9=SWAP   F10=LEFT   F11=RIGHT  F12=RETRIEVE
```

Figure 3-55 SDSF: Options menu

Before choosing any option, place the cursor in the menu action bar, select **Options**, and press Enter. Option 5 shows Set display values to OFF. If you then choose this option, the display options value will be set to OFF. You get the same result by issuing the SDSF command **SET DISPLAY OFF** in the COMMAND INPUT line.

You can customize your SDSF panels by choosing the **View** option in the action bar and then the **Arrange** option. You do that for each panel where the **Arrange** option is available and choose the fields in the order you want them. SDSF stores the information in your ISPF/PDF profile data set and uses them the next time you enter any SDSF options.

## 3.56 SDSF: Viewing the JES2 output files

```

Screen 1

  Display Filter View Print Options Help
-----
SDSF HELD OUTPUT DISPLAY ALL CLASSES LINES 44          LINE 1-1 (1)
COMMAND INPUT ==>                                     SCROLL ==> PAGE
PREFIX=* DEST=(ALL) OWNER=* SYSNAME=
NP  JOBNAME JobID  Owner  Prty C ODisp Dest          Tot-Rec Tot-
?_  MIRIAM2  JOB26044 MIRIAM  144 T HOLD LOCAL          44      44

Screen 2

  Display Filter View Print Options Help
-----
SDSF JOB DATA SET DISPLAY - JOB MIRIAM2 (JOB26044)  LINE 1-3 (3)
COMMAND INPUT ==>                                     SCROLL ==> PAGE
PREFIX=* DEST=(ALL) OWNER=* SYSNAME=
NP  DDNAME  StepName ProcStep DSID Owner  C Dest          Rec-Cnt Page
    JESMSGLG JES2          2 MIRIAM T LOCAL          20
    JESJCL   JES2          3 MIRIAM T LOCAL          12
    JESYSMSG JES2          4 MIRIAM T LOCAL          12

```

Figure 3-56 SDSF: Viewing the JES2 output files

You can see the JES output data sets created during the execution of your batch job. They are saved on the JES spool data set. You can see the JES data sets in any JES queue: input (i SDSF option), execution queue (DA option), output queue (O option) or held queue (H option).

For output and held queues, you cannot see those JES data sets you requested to be automatically purged by setting a MSGCLASS out sysout CLASS that has been defined to not save output. Also, depending on the MSGCLASS you chose in the JOB card, the sysouts can be in the Output queue or in the Held queue.

The first screen shown in Figure 3-56 displays a list of the jobs we submitted and whose output we directed to the HELD (Class T) queue, as identified in the MSGCLASS=T parameter in the job card. In our case only one job has been submitted and executed. Therefore, we only have one job on the held queue.

Issuing a ? command in the NP column displays the output files generated by job 7359. The second screen shown in Figure 3-56 displays three ddnames: JES2 messages log file, JES2 JCL file, and JES2 system messages file. This option is useful when you are seeing jobs with many files directed to SYSOUT and you want to display one associated with a specific step. You issue an S in the NP column to select a file you want.

To see all files, instead of a ?, type S in the NP column; the result is presented in the next figure.

```

J E S 2  J O B  L O G  --  S Y S T E M  S C 6 4  --  N O D E

13.19.24 JOB26044 ---- WEDNESDAY, 27 AUG 2003 ----
13.19.24 JOB26044 IRR010I USERID MIRIAM IS ASSIGNED TO THIS JOB.
13.19.24 JOB26044 ICH70001I MIRIAM LAST ACCESS AT 13:18:53 ON WEDNESDAY, AUGU
13.19.24 JOB26044 $HASP373 MIRIAM2 STARTED - INIT 1 - CLASS A - SYS SC64
13.19.24 JOB26044 IEF403I MIRIAM2 - STARTED - ASID=0027 - SC64
13.19.24 JOB26044 - --TIMINGS (MINS.)--
13.19.24 JOB26044 -JOBNAME STEPNAME PROCSTEP RC EXCP CPU SRB CLOCK
13.19.24 JOB26044 -MIRIAM2 STEP1 00 9 .00 .00 .00
13.19.24 JOB26044 IEF404I MIRIAM2 - ENDED - ASID=0027 - SC64
13.19.24 JOB26044 -MIRIAM2 ENDED. NAME=MIRIAM TOTAL CPU TIME=
13.19.24 JOB26044 $HASP395 MIRIAM2 ENDED
----- JES2 JOB STATISTICS -----
27 AUG 2003 JOB EXECUTION DATE
11 CARDS READ
44 SYSOUT PRINT RECORDS
0 SYSOUT PUNCH RECORDS
3 SYSOUT SPOOL KBYTES
0.00 MINUTES EXECUTION TIME
1 //MIRIAM2 JOB 19,MIRIAM,NOTIFY=&SYSUID,MSGCLASS=T,
// MSGLEVEL=(1,1),CLASS=A
IEFC653I SUBSTITUTION JCL - 19,MIRIAM,NOTIFY=MIRIAM,MSGCLASS=T,MSGLEVE
2 //STEP1 EXEC PGM=IEFBR14
/*-----*
/** THIS IS AN EXAMPLE OF A NEW DATA SET ALLOCATION
/*-----*
3 //NEWDD DD DSN=MIRIAM.IEFBR14.TEST1.NEWDD,
// DISP=(NEW,CATLG,DELETE),UNIT=SYSDA,
// SPACE=(CYL,(10,10,45)),LRECL=80,BLKSIZE=3120
4 //SYSPRINT DD SYSOUT=T
/*
ICH70001I MIRIAM LAST ACCESS AT 13:18:53 ON WEDNESDAY, AUGUST 27, 2003
IEF236I ALLOC. FOR MIRIAM2 STEP1
IGD100I 390D ALLOCATED TO DDNAME NEWDD DATACLAS ( )
IEF237I JES2 ALLOCATED TO SYSPRINT
IEF142I MIRIAM2 STEP1 - STEP WAS EXECUTED - COND CODE 0000
IEF285I MIRIAM.IEFBR14.TEST1.NEWDD CATALOGED
IEF285I VOL SER NOS= SBOX38.
IEF285I MIRIAM.MIRIAM2.JOB26044.D0000101.? SYSOUT
IEF373I STEP/STEP1 /START 2003239.1319
IEF374I STEP/STEP1 /STOP 2003239.1319 CPU OMIN 00.00SEC SRB OMIN 00.00S
IEF375I JOB/MIRIAM2 /START 2003239.1319
IEF376I JOB/MIRIAM2 /STOP 2003239.1319 CPU OMIN 00.00SEC SRB OMIN 00.00S

```

*SDSF displaying the output job*

This figure shows the output for our job. The most important things to note are:

1. The RC or Return Code value is 00 which indicates a successful completion of the step.
2. The COND CODE or Condition Code is 0000 which indicates a successful completion of the job.
3. The following messages show that the data set MIRIAM.IEFBR14.TEST.NEWDD was allocated on disk volume SBOXA7, and was cataloged.

```

IEF285I MIRIAM.IEFBR14.TEST.NEWDD CATALOGED
IEF285I VOL SER NOS= SBOXA7

```

## 3.57 SDSF: Display Active Users (DA)

```

Display Filter View Print Options Help
-----
SDSF DA SC67 SC67 PAG 0 SIO 7 CPU 6/ 7 LINE 1-25 (64)
COMMAND INPUT ==> SCROLL ==> PAG
PREFIX=* DEST=LOCAL OWNER=* SORT=JOBNAME/A
NP JOBNAME STEPNAME PROCSTEP JOBID OWNER C POS DP REAL PAGING SIO
 *MASTER* STC06373 +MASTER+ NS FF 1369 0.00 0.00
 ALLOCAS ALLOCAS NS FF 190 0.00 0.00
 ANTAS000 ANTAS000 IEFPROC NS FE 1216 0.00 0.00
 ANTMAIN ANTMAIN IEFPROC NS FF 4541 0.00 0.00
 APPC APPC APPC NS FE 2653 0.00 0.00
 ASCH ASCH ASCH NS FE 267 0.00 0.00
 BPXOINIT BPXOINIT BPXOINIT LO FF 315 0.00 0.00
 CATALOG CATALOG IEFPROC NS FF 1246 0.00 0.00
 CICSPPAY CICSPPAY CICS520 STC06504 STC NS FE 4330 0.00 0.00
 CONSOLE CONSOLE NS FF 597 0.00 0.00
 DFRMM DFRMM IEFPROC STC06363 STC NS FE 510 0.00 0.00
 DFSMSHSM HSMSC67 DFSMSHSM STC13178 STC NS FE 6199 0.00 0.00
 DUMPSRV DUMPSRV DUMPSRV NS FF 160 0.00 0.00
 FTPDMVS1 STEP1 STC06477 STC LO FF 470 0.00 0.00
 FTPDOE1 STEP1 STC06475 FTPDOE LO FF 469 0.00 0.00
 GRS GRS NS FF 894 0.00 0.00
 IEFSCHAS IEFSCHAS NS FF 25 0.00 0.00
 IMWEBSUF IMWEBSUF WEBSRV STC15245 WEBSRV IN FE 15T 0.00 0.00

```

Figure 3-57 SDSF - Display Active Users (DA)

SDSF provides the ability to monitor the current system workload. The **DA** command displays the active tasks and provides information about each task. This information includes CPU usage for each task, the amount of CPU time that a task has used, and the Input/Output related EXCP statistics. Figure 3-57 displays some of the data that this facility captures. Press PF11 to move to the right and see all the available fields.

When you command **DA OTSU**, SDSF displays only TSO users, **DA OJOB** shows only the JES jobs running, and **DA OSTC** shows only the active started tasks.

SDSF provides action commands you can use in the **NP** column. For example, from the Active Users panel, a user can enter **S** in the **NP** column next to a job to look at the output data set for that job. SDSF displays the output data set on the Output Data Set panel. You may not have authority to issue some of the available commands. In this case, SDSF issues a message in the top right of the panel. When you choose an action command, SDSF issues the system command that corresponds to the action you chose.

To display which action commands can be used in an SDSF panel, issue the **HELP** command in each option panel. Then choose option **3 - Action characters**. A panel listing all the action commands you can use in that option appears.

## 3.58 Issuing MVS and JES commands

```
Display Filter View Print Options Help
-----
HQX7707 ----- SDSF PRIMARY OPTION MENU -- PARM INVALID
COMMAND INPUT ==> /SET PROG+ SCROLL ==> PAGE

DA      System Command Extension
I
O      Type or complete typing a system command, then press Enter.
H
ST      ==> SET PROG
      ==> _____

LO      Place the cursor on a command and press Enter to retrieve it.
SR
MA      More:      +
JC      => D T
SE      => CANCEL U=ORSI
RE      => SET PROG
EN      =>
PS      =>
EN      =>
      =>
      F1=Help      F2=Split      F3=Cancel      F5=FullScr      F7=Backward
      F8=Forward   F9=Swap      F11=ClearLst  F12=Cancel
```

Figure 3-58 Using SDSF to issued MVS and JES Commands

If you are authorized, on the COMMAND INPUT ==> line of any SDSF panel you can issue any MVS or JES2 command following a slash (/). If the command is too long, type + and two more lines will appear to allow you to type the rest of the command. If you have authority, you can use the **ULOG** option to see only your commands and their response.

**Proceed with caution in the SDSF DA panel:** If you issue a **C** (Cancel) action command in the DA display it will cancel any task you select if you have the authority. Use this command with caution if you are displaying major production tasks. Commands issued in the DA display are issued against running tasks. Incorrect or careless use can cause major problems.

### 3.59 SDSF: Input queue panel

```
Display Filter View Print Options Help
-----
SDSF INPUT QUEUE DISPLAY ALL CLASSES LINE 1-7 (7)
COMMAND INPUT ==> SCROLL ==> PAGE
PREFIX=* DEST=(ALL) OWNER=* SYSNAME=
NP JOBNAME JobID Owner Prty C Pos PrtDest Rmt Node SAF
BARTR1DB JOB06472 BARTR1 10 A LOCAL 1
BARTR1DB JOB06479 BARTR1 10 A LOCAL 1
BARTR1DB JOB06561 BARTR1 10 A LOCAL 1
BARTR1DB JOB06565 BARTR1 10 A LOCAL 1
BARTR1DB JOB06568 BARTR1 10 A LOCAL 1
BARTR1DB JOB06588 BARTR1 10 A LOCAL 1
BARTTEP1 JOB09138 BART 10 A LOCAL 1 SC6

F1=HELP F2=SPLIT F3=END F4=RETURN F5=IFIND F6=B00K
F7=UP F8=DOWN F9=SWAP F10=LEFT F11=RIGHT F12=RETRIEVE
```

Figure 3-59 SDSF: Input queue panel

The input queue panel provides information about jobs, TSO users, and started tasks that are on the JES2 input queue or are being processed by the system. Users display this information by entering **i** on the command input line of any SDSF panel. An example of this panel is shown in Figure 3-59.

This figure shows only a subset of the fields of information that are available on the Input Queue panel. You can scroll right, left, up, and down throughout this panel. Also you can control the scroll using the SCROLL ==> field. You can use, for example, **C** (CSR), to better control the scroll.

## 3.60 SDSF: Output queue panel

```

Display Filter View Print Options Help
-----
SDSF OUTPUT ALL CLASSES ALL FORMS      LINES 304,174      LINE 1-24 (266)
COMMAND INPUT ==>                          SCROLL ==> PAGE
PREFIX=* DEST=(ALL) OWNER=* SYSNAME=
NP  JOBNAME  JobID  Owner  Prty C Forms  Dest  Tot-Rec
   RMF      STC16499  STC      144 A STD   LOCAL
   JJONESDB JOB17936  JJONES   144 A STD   LOCAL          34
   JJONESDB JOB17937  JJONES   144 A STD   LOCAL         145
   RMF      STC17097  STC      144 A STD   LOCAL
   RMF      STC18679  STC      144 A STD   LOCAL
   RMF      STC13665  STC      144 A STD   LOCAL
   LUTZ     TSU20005  LUTZ     144 A STD   LOCAL          24
   LUTZ     TSU20206  LUTZ     144 A STD   LOCAL          24
   LUTZ     TSU20555  LUTZ     144 A STD   LOCAL          24
   ARS01X   JOB20692  TWSRES1  144 A STD   LOCAL          29
   ARS01X   JOB20693  TWSRES1  144 A STD   LOCAL          29
   ARS01X   JOB20717  TWSRES1  144 A STD   LOCAL          29
   LDAPKI   STC19980  LDAPKI   144 A STD   LOCAL          54
   RMF      STC19444  STC      144 A STD   LOCAL
   HSM      STC21908  STC      144 A STD   LOCAL          19
   HSM      STC21908  STC      144 A STD   LOCAL          18
   HSM      STC21908  STC      144 A STD   LOCAL          19
   HSM      STC21908  STC      144 A STD   LOCAL          19
   HSM      STC21908  STC      144 A STD   LOCAL           2
   HSM      STC21908  STC      144 A STD   LOCAL           2
   HSM      STC21908  STC      144 A STD   LOCAL           2
   TWS      JOB22149  VBUDI    144 A STD   LOCAL         354
   TWS      JOB22151  VBUDI    144 A STD   LOCAL         375
   TWS      JOB22153  VBUDI    144 A STD   LOCAL         101
F1=HELP  F2=SPLIT  F3=END    F4=RETURN  F5=IFIND  F6=B00K
F7=UP    F8=DOWN   F9=SWAP   F10=LEFT   F11=RIGHT  F12=RETRIEVE

```

Figure 3-60 SDSF: Output queue panel

The SDSF **O** command displays the *non-held output queue*. The Output Queue panel provides information about output data sets for jobs, TSO users, and started tasks that are on a JES2 output queue. Users display this information by entering **O** on the command input line of any SDSF panel. An example of this panel is shown in Figure 3-60.

You might want to use the **PREFIX** command to limit the entries in the display. For example,

- ▶ **PREFIX**, with no parameter displays all jobs for which you are authorized.
- ▶ **PREFIX MQ\*** displays all jobs that start with the name MQ.
- ▶ **PREFIX M%R\*** displays all jobs that begin with M and have R in the fourth position.

The default, when you first enter SDSF, is your logon ID prefix. If SDSF is not displaying what you expect, issue the **SET DISPLAY ON** command. This controls the display of values you have set for PREFIX, DEST, OWNER, SORT, and FILTER.

The following is an output display using **prefix DB2\***.

NP	JOBNAME	StepName	ProcStep	JobID	Owner	C	Pos	DP	Real	Paging	SIO
	DB2GDBM1	DB2GDBM1	IEFPROC	STC04424	STC		NS	FE	9826	0.00	0.00
	DB2GDIST	DB2GDIST	IEFPROC	STC04425	STC		NS	FE	2313	0.00	0.00
	DB2GIRLM	DB2GIRLM		STC04423	STC		NS	FE	284	0.00	0.00
	DB2GMSTR	DB2GMSTR	IEFPROC	STC04422	STC		NS	FE	1887	0.00	0.00
	DB2GSPAS	DB2GSPAS	IEFPROC	STC04426	STC		NS	FB	982	0.00	0.00



### 3.61 SDSF: Held Output queue panel

```
Display Filter View Print Options Help
-----
SDSF HELD OUTPUT DISPLAY ALL CLASSES LINES 194 LINE 1-6 (6)
COMMAND INPUT ==> SCROLL ==> PAGE
PREFIX=* DEST=(ALL) OWNER=* SYSNAME=
NP JOBNAME JobID Owner Prty C ODisp Dest Tot-Rec Tot-
MIRIAM2 JOB26044 MIRIAM 144 T HOLD LOCAL 44
MIRIAM2 JOB26069 MIRIAM 144 T HOLD LOCAL 30
MIRIAM3 JOB26070 MIRIAM 144 T HOLD LOCAL 30
MIRIAM4 JOB26071 MIRIAM 144 T HOLD LOCAL 30
MIRIAM5 JOB26072 MIRIAM 144 T HOLD LOCAL 30
MIRIAM6 JOB26073 MIRIAM 144 T HOLD LOCAL 30

F1=HELP F2=SPLIT F3=END F4=RETURN F5=IFIND F6=B00K
F7=UP F8=DOWN F9=SWAP F10=LEFT F11=RIGHT F12=RETRIEVE
```

Figure 3-61 SDSF: Held Output queue panel

The Held Output queue panel provides information about output that is being held on any JES2 output queue. Users display this information by entering **h** on the command input line of any SDSF panel. An example of this panel is shown in Figure 3-61. This figure shows only a subset of the information available on the Held Output queue panel. From this panel, a user can look at the output for a specific job and then decide to print or purge the output.

## 3.62 SDSF: Status panel

```

Display Filter View Print Options Help
-----
SDSF STATUS DISPLAY ALL CLASSES                               LINE 1-24 (3281)
COMMAND INPUT ==>                                           SCROLL ==> PAGE
PREFIX=* DEST=(ALL) OWNER=* SYSNAME=
NP  JOBNAME JobID  Owner  Prty Queue  C  Pos  SAff  ASys  Status
BARTR1DB JOB06472 BARTR1  10 EXECUTION A          HOLD
BARTR1DB JOB06479 BARTR1  10 EXECUTION A          HOLD
BARTR1DB JOB06561 BARTR1  10 EXECUTION A          HOLD
BARTR1DB JOB06565 BARTR1  10 EXECUTION A          HOLD
BARTR1DB JOB06568 BARTR1  10 EXECUTION A          HOLD
BARTR1DB JOB06588 BARTR1  10 EXECUTION A          HOLD
BARTTEP1 JOB09138 BART   10 EXECUTION A          HOLD
TWSSTD3  TSU26002 TWSSTD3  15 EXECUTION SC63 SC64
KMT1     TSU26024 KMT1    15 EXECUTION SC64 SC64
MIRIAM   TSU26043 MIRIAM   15 EXECUTION SC64 SC64
HAIMO    TSU26050 HAIMO    15 EXECUTION SC63 SC63
BARTR4   TSU26051 BARTR4   15 EXECUTION SC63 SC63
RAVI     TSU26052 RAVI     15 EXECUTION SC63 SC63
BARTR2   TSU26060 BARTR2   15 EXECUTION SC63 SC63
VBUDI    TSU26062 VBUDI    15 EXECUTION SC64 SC64
SYSLOG   STC24863 +MASTER+ 15 EXECUTION SC63 SC63
RACF     STC24871 RACF     15 EXECUTION SC63 SC63
SYSLOG   STC24931 +MASTER+ 15 EXECUTION SC64 SC64
RACF     STC24941 RACF     15 EXECUTION SC64 SC64
OPTSO    STC24857 STC      15 EXECUTION SC63 SC63
OAM      STC24858 STC      15 EXECUTION SC63 SC63
RMF      STC24855 STC      15 EXECUTION SC63 SC63
SDSF     STC24862 STC      15 EXECUTION SC63 SC63 ARMELEM
ASCHINT  STC24867 STC      15 EXECUTION SC63 SC63
F1=HELP  F2=SPLIT  F3=END    F4=RETURN  F5=IFIND  F6=B00K
F7=UP    F8=DOWN   F9=SWAP   F10=LEFT   F11=RIGHT F12=RETRIEVE

```

Figure 3-62 SDSF: Status panel

The Status panel provides information about jobs, started tasks, and TSO users that are on any of the JES2 queues. Users display this information by entering `st` on the command input line of any SDSF panel. An example of this panel is shown in Figure 3-62. This figure shows only a subset of the fields of information available on the Status panel.

### 3.63 SDSF: Tutorial

```
Display Filter View Print Options Help
      TUTOR - System Display and Search Facility
COMMAND INPUT ==> _

The SDSF tutorial introduces SDSF and lets you
try some of SDSF's most useful functions. For detailed
information such as command syntax, use the help facility.

The whole tutorial takes about 25 minutes. Press Enter to
begin viewing it, or begin with a particular topic by
typing one of the numbers below:

      1 - Using the tutorial      5 - Purging output
      2 - SDSF panels            6 - Controlling jobs
      3 - Monitoring jobs        7 - Printing data
      4 - Displaying output      8 - Filtering and sorting

      9 - Quick summary

      F1=Help      F2=Split      F5=Exhelp      F7=Up      F9=Swap
      F10=Previous F11=Index    F12=Cancel

F1=HELP      F2=SPLIT      F3=END      F4=RETURN      F5=IFIND      F6=B00K
F7=UP        F8=DOWN      F9=SWAP      F10=LEFT      F11=RIGHT      F12=RETRIEVE
```

Figure 3-63 SDSF: Tutorial

You can use the Tutorial Panel to get familiar with all the SDSF functions. This panel is displayed when you Enter the command **tutor** on the command line.

The tutorial shows you some basic functions of SDSF. Some parts of the tutorial ask you to enter information on simulated SDSF panels. These simulated panels respond to your input. Interacting with the simulated panels will help you learn how SDSF works. However, if you prefer, the system will provide the input on interactive panels if you simply press Enter twice. Except on the interactive tutorial panels, SDSF commands are not valid on tutorial or help panels.





## MVS delivery and installation

z/OS consists of base elements and optional features. The base elements deliver essential operating system functions. The optional features are orderable with z/OS and provide additional operating system functions.

Because the base elements and optional features of z/OS are integrated into a single package with compatible service levels, you must install, with few exceptions, the entire z/OS product. You can install z/OS using one of several IBM packages.

This chapter presents the available z/OS delivery options to install z/OS, as follows:

- ▶ ServerPac
- ▶ CBPDO
- ▶ SystemPac
- ▶ SoftwareXcel
- ▶ Entry Server Offering

In addition, this chapter describes briefly the download process and installation steps for using the ServerPac installation option.

**Note:** When you order any one of the installation packages, you will receive a comprehensive installation guide, detailing the installation tasks step-by-step from the beginning of the installation until you IPL your system. For example, if you choose the ServerPac installation package, you receive the *ServerPac: Installing Your Order* documentation that is tailored to your order for installation. This document is unique to your environment and is based on what you have ordered.

## 4.1 z/OS installation overview

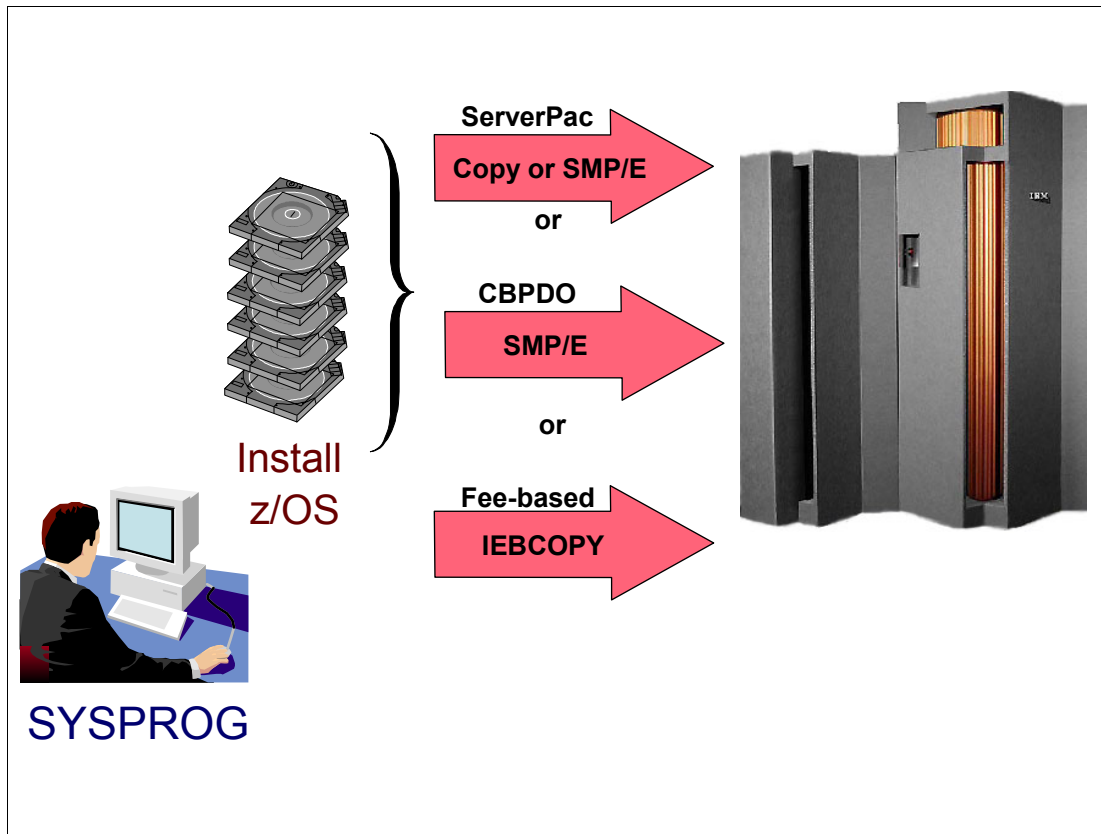


Figure 4-1 System programmer installing z/OS

Prior to z/OS, large applications ran on an MVS operating system that consisted of the Basic Control Program (BCP), the Data Facility Product (DFSMSdfp), and Job Entry Subsystem (JES2 or JES3), plus a collection of other software products that the applications required, such as Interactive System Productivity Facility (ISPF), Time Sharing Option/Extensions (TSO/E), and so forth. You traditionally ran these products at various release levels, using a "mix and match" approach.

With the introduction of z/OS, all these products were integrated into a single product. You no longer order new levels of some products but not of others; instead, you order and install an entire set of products integrated into one functionally rich operating system.

## 4.2 z/OS release cycle

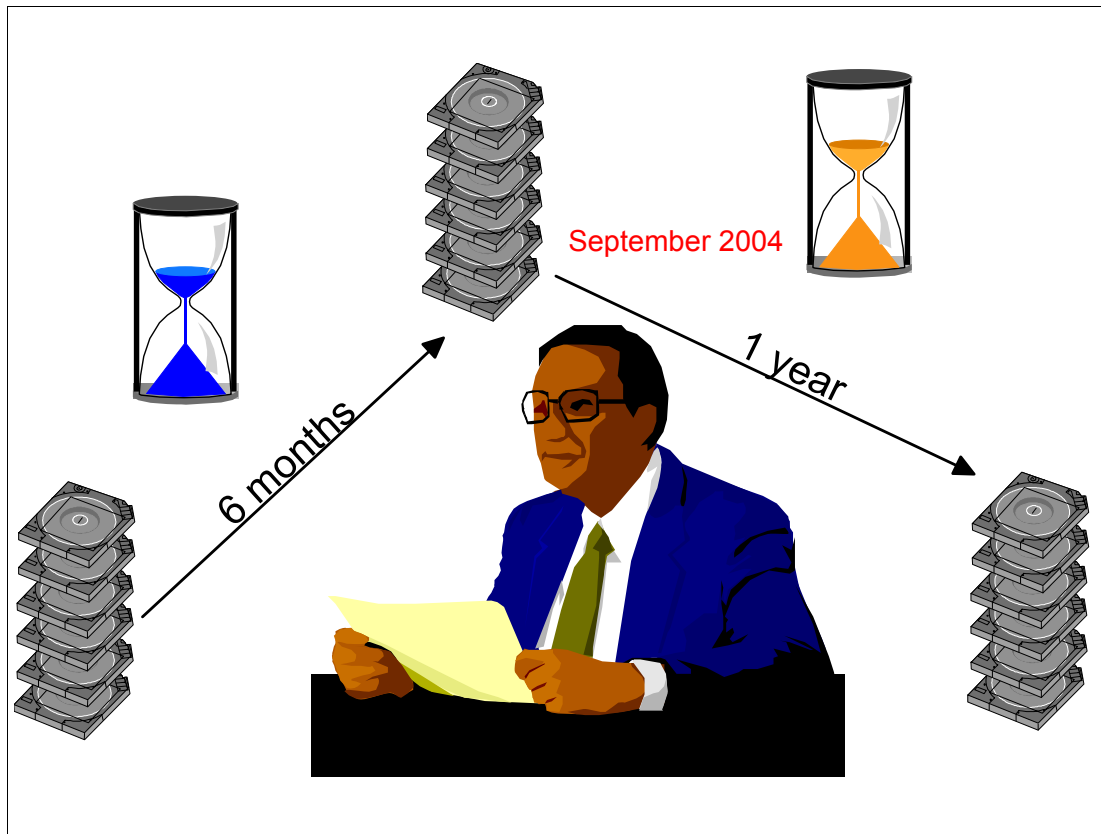


Figure 4-2 z/OS Release Cycle

The release cycle of z/OS is approximately every six months, hence you expect to receive function in new levels from every release. At the end of every March and September, a new release becomes generally available, with shipments starting at the beginning of April and October. Note that not every element and feature in a given release contains new function. However, those elements and features that don't receive new function have all eligible service included.

This predictable release cycle should reduce your planning time. Because each new level is comprehensively tested, the quality of the operating system is improved. Once your initial migration is complete, you can expect simplified ordering, planning, and installing of the next z/OS release.

**Note:** Beginning in September 2004, IBM will change the release cycle of z/OS to an annual basis, with shipping of a new release of z/OS each September.

## 4.3 z/OS delivery options

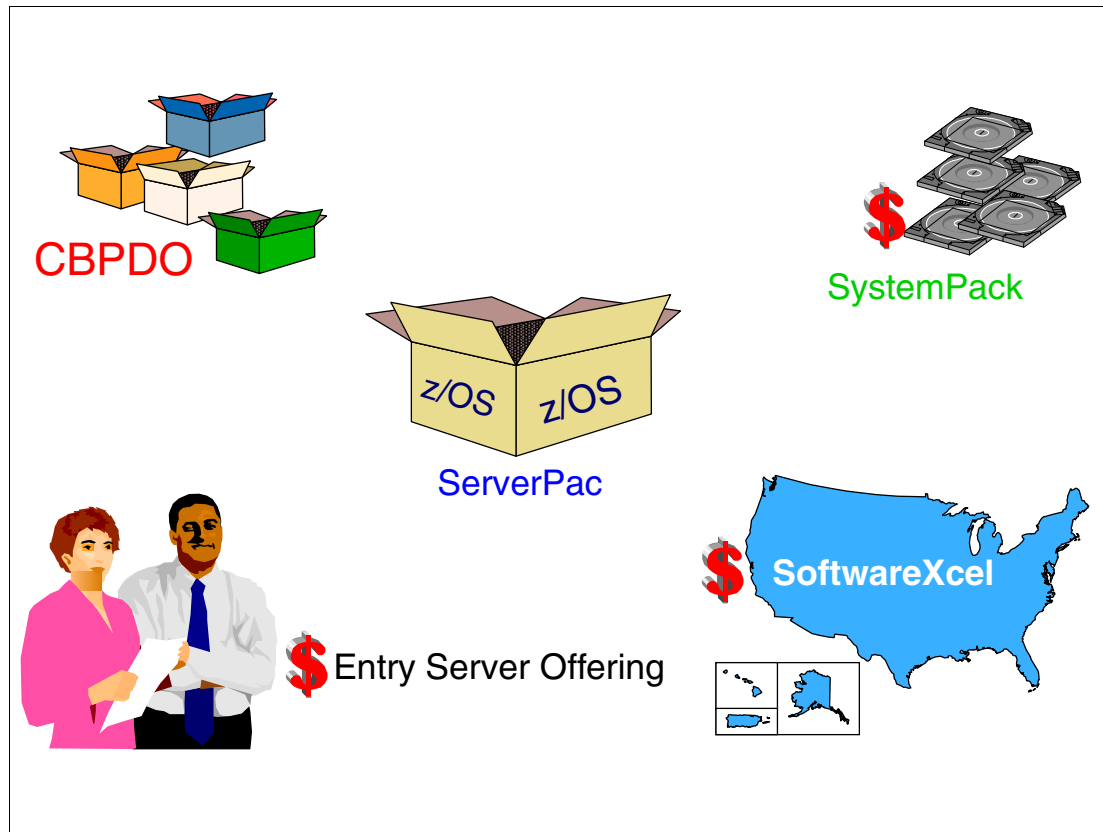


Figure 4-3 z/OS Package Options

You can install z/OS using one of several IBM packages. These two packages are available at no additional charge when you license z/OS:

1. **ServerPac:** This software delivery package consists of products and service for which IBM has performed the SMP/E installation steps and some of the post-SMP/E installation steps. You use the CustomPac Installation Dialog to install your system and complete the installation of the software it includes.
2. **CBPDO:** The Custom Built Product Delivery Option (CBPDO) is a delivery package that consists of uninstalled products and unintegrated service. You must use SMP/E to install the individual z/OS elements and features, and their service, before you can IPL.

Several fee-based options are available, including:

3. **SystemPac:** SystemPac offers the capability to build a system with integrated subsystems in either full volume dump/restore format or data set copy format. The full volume dump/restore format enables you to install z/OS without using the dialog. Installation is done via pack restore using DFSMSdss or FDR (if the vendor product is selected in the order).

SystemPac is designed for those who have limited skill or time to install or upgrade z/OS, but who want to install or upgrade to exploit z/OS functions in e-commerce or other areas. SystemPac tailors z/OS to your environment (such as DASD layout, migration of MSVCP/IOCP to IODF, and naming conventions) based on information provided to IBM. With this offering, selected non-IBM products can be integrated.



4. **SoftwareXcel:** SoftwareXcel Installation Express (SIE) is available in the U.S. only, and provides pre-built z/OS system packages in full volume dump format, tailored to customer hardware and software configurations. SIE includes on-site planning, installation, and package testing.
5. **Entry Server Offering:** The Entry Server Offering, only available in selected countries, is a packaged solution that includes hardware, software, installation services, maintenance, and financing to help customers get current technology.

## 4.4 ServerPac service level

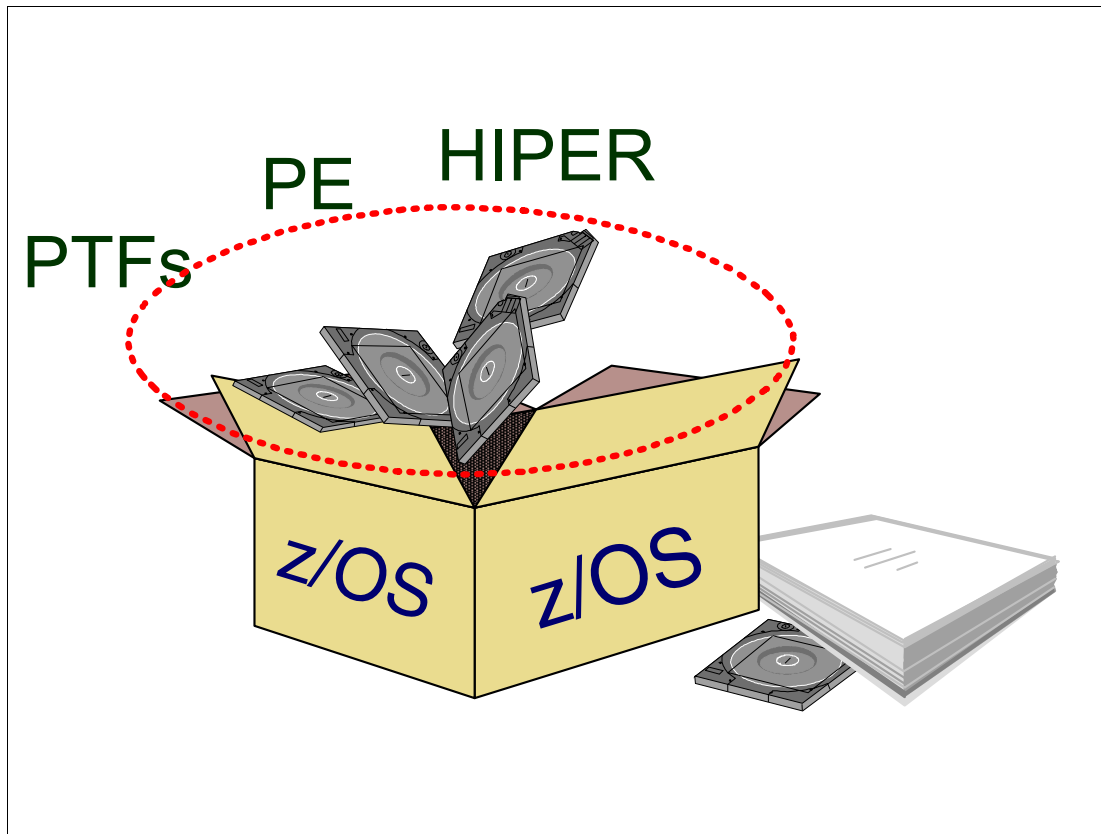


Figure 4-4 ServerPac service level

For ServerPac orders, service is *integrated* with product code according to the following time-line:

- ▶ ServerPac is refreshed every month to pick up the most current Recommended Service Upgrades (RSUs).
- ▶ All products incorporate HIPER and PTF-in-error (PE) fixes that are available approximately one week before your order is received.
- ▶ Your ServerPac order also contains a service tape containing all unintegrated service that was available at the time your order was processed.

## 4.5 CBPDO service level

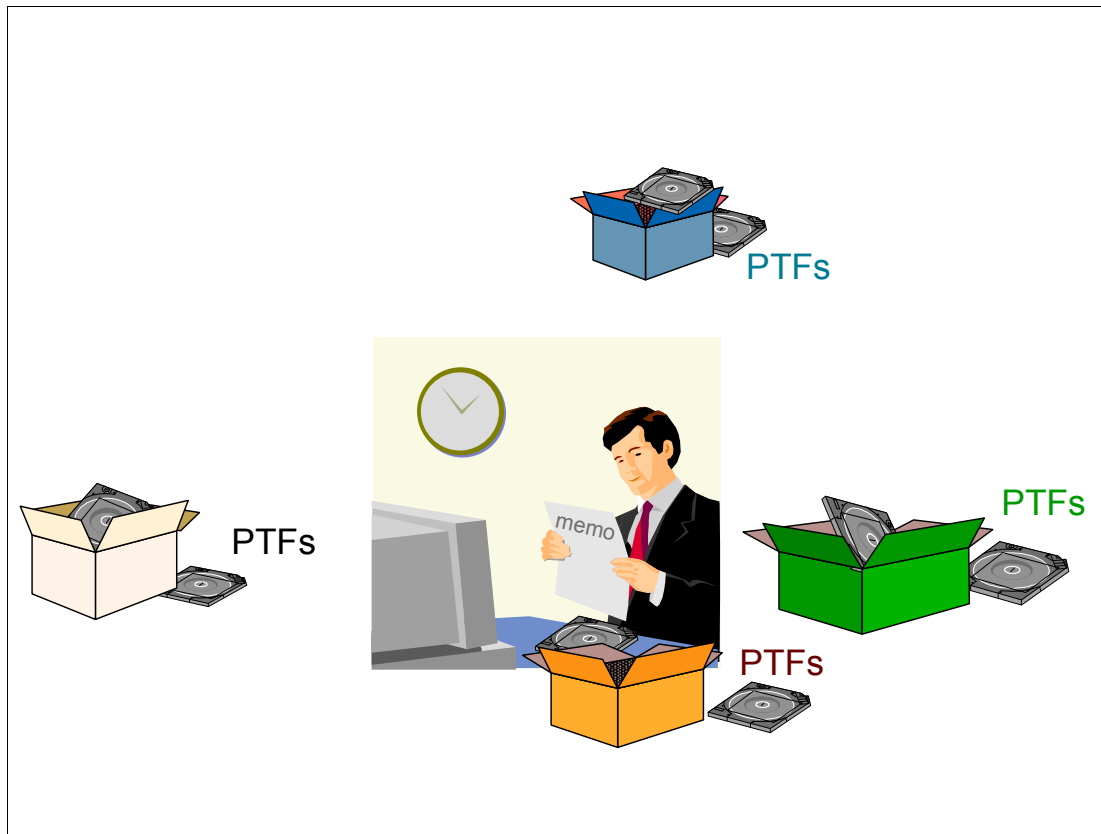


Figure 4-5 CBPDO service level

Remember, the service in the CBPDO orders is *not integrated*. You must receive and apply the service during the installation process. The service that is shipped with every CBPDO order is as follows:

- ▶ Service for all products previously ordered, as reflected in the customer profile for the customer number used when the order was placed. You can specify the starting date for these PTFs at order time. The default starting date is the last time that customer number was used to order a CBPDO (product or service) for that system release identifier (SREL).
- ▶ Service for all products, elements, and features that you have ordered. This includes all PTFs that became available between product general availability and the time of your order that have not been incorporated in the product FMIDs.

**Note:** A *Memo to Users Extension* comes with CBPDO; it describes the source IDs for service delivered on the CBPDO tape.

## 4.6 System and installation requirements

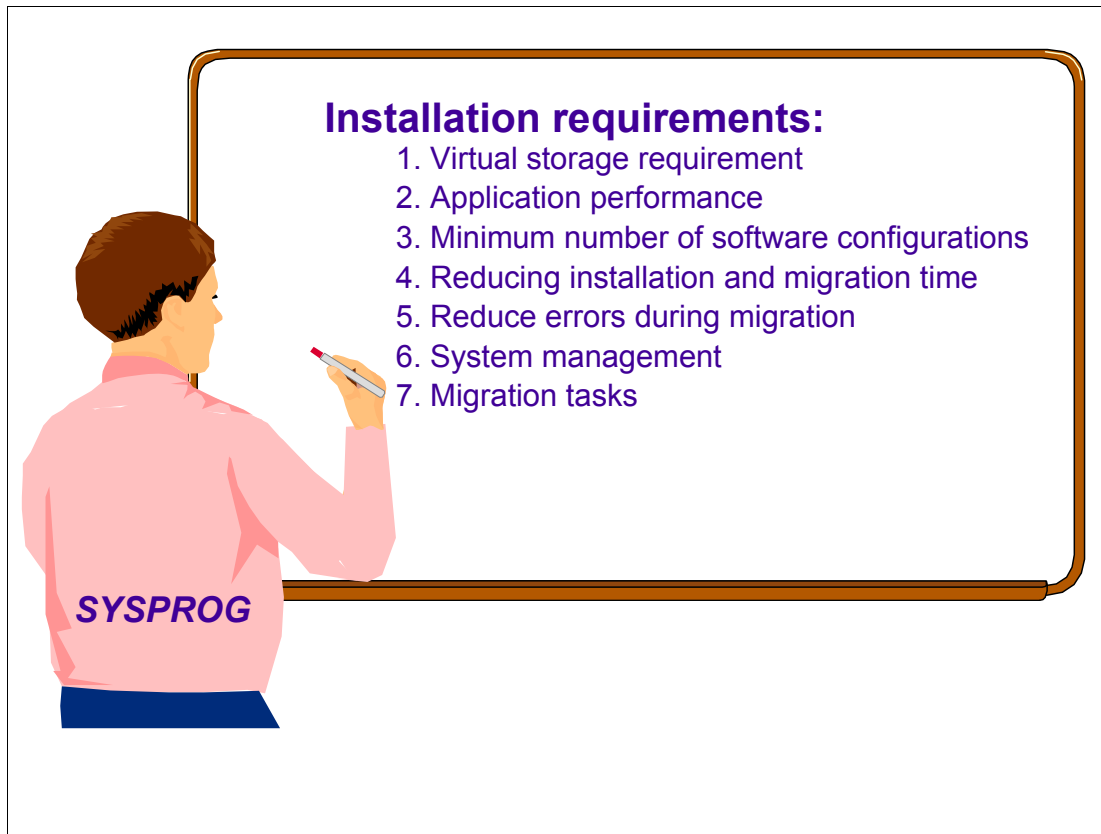


Figure 4-6 Determining the installation requirements

Having an installation plan helps you ensure the software is able to meet your installation's functional requirements. Besides functionality, there are other issues to think about when planning to build a system. These additional considerations include:

- ▶ Hardware and software requirements, including non-IBM software compatibility
- ▶ Virtual storage mapping
- ▶ Application performance
- ▶ Building a minimum number of system software configurations
- ▶ Reducing installation and migration time
- ▶ Reducing the opportunities for error during migration
- ▶ Making it easier to manage the system after it is in production
- ▶ Minimizing migration actions for the people who use the system

How you choose to meet all these requirements can have a significant effect on how much work is required to perform the tasks associated with each stage. Keep all these additional requirements in mind when you are planning to build a new system.

## 4.7 Reviewing your current system

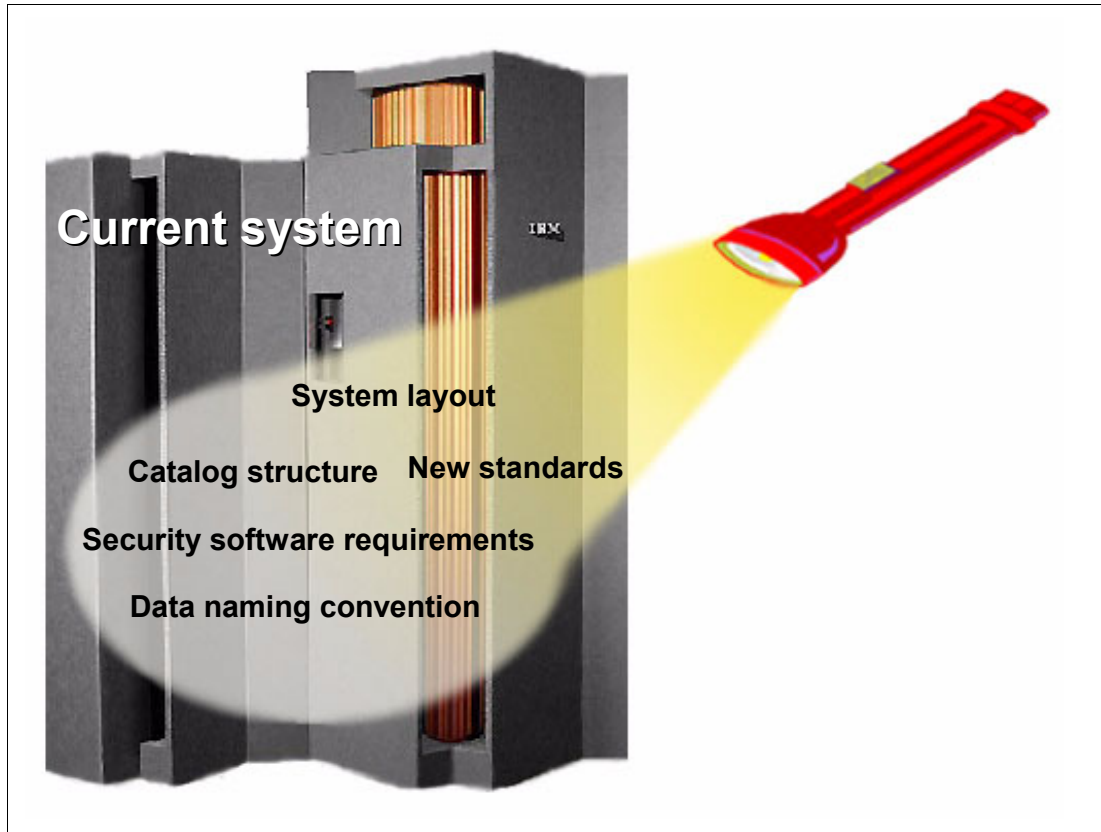


Figure 4-7 Reviewing your current system for migration

More often than not, you are planning to migrate from your current MVS system to the new release of z/OS. It is therefore very important to review the setup of your current environment while planning for the new system. Some of the things you should consider are:

- ▶ The system layout
- ▶ The catalog structure
- ▶ Data set naming conventions in your present environment
- ▶ Security software considerations
- ▶ New standards, if necessary

Depending on your order, the system target and distribution libraries may exceed more than one DASD volume, for example IBM 3390-3. You should define your new system layout to be prepared for future installation and easy cloning of your system. We recommend that you read “Recommended data set placement” on *z/OS V1R4.0 and z/OS.e V1R4.0 Planning for Installation*, GA22-7504 before defining where the following new data sets should reside:

- ▶ Target data sets
- ▶ Distribution libraries data sets
- ▶ Master catalog and user catalogs
- ▶ Dialog and order data sets

## 4.8 The driving and target system

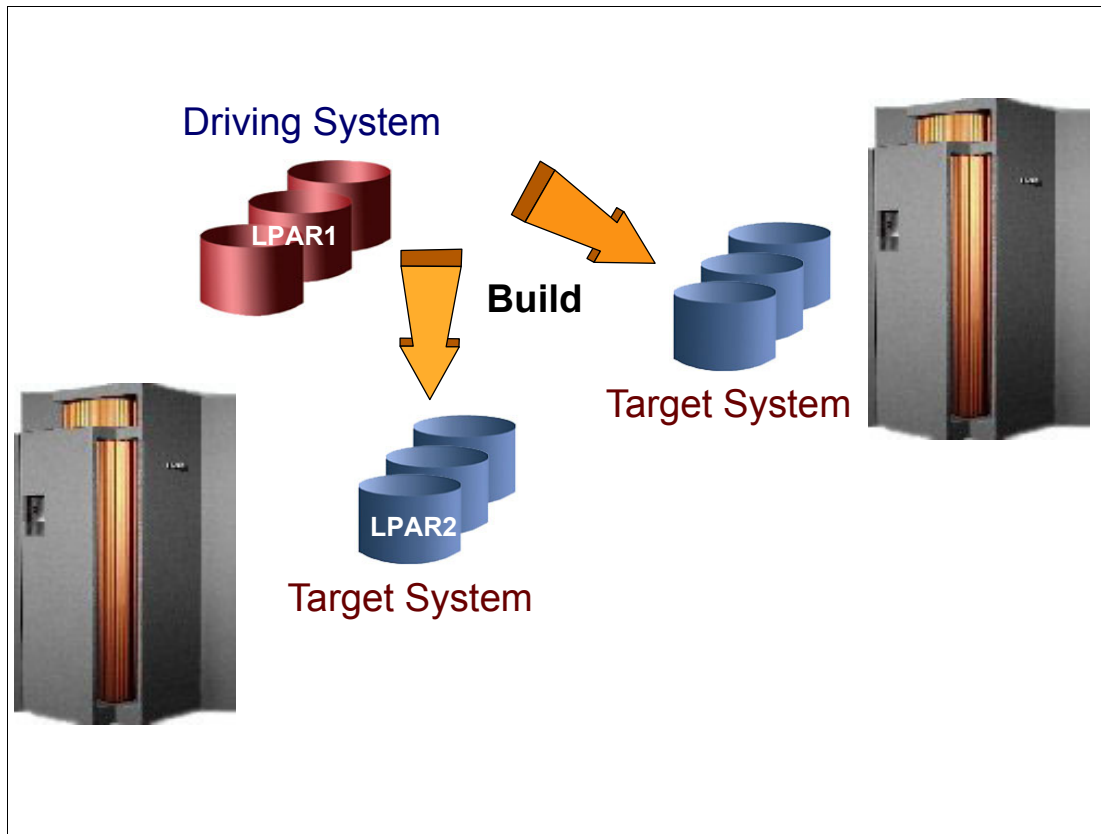


Figure 4-8 Using the driving system to create the new system

The *driving system* is the system image (both the hardware and software) that you use to install your new system image, the *target system*. You log on to the driving system and run jobs there to create or update the target system. Once the target system is built, it can be IPLed on the same hardware (same LPAR or different LPAR on the same processor) or on different hardware than that used for the driving system.

To prepare the driving system *before* building the target system, you need to perform the following tasks:

- ▶ Identify the software requirements for the driving system according to the delivery package you are using, for example, ServerPac.
- ▶ Identify the hardware requirements for the driving system.

You are also required to do some preparations for the target system:

- ▶ Choose the software products to install and identify requisites.
- ▶ Order z/OS and related IBM products.
- ▶ Identify the hardware requirements for the target system.
- ▶ Identify the service needed for the target system.
- ▶ Decide whether to use the existing JES2 or JES3 with the new z/OS release.

For more information on each of the requirements for both the driving and target system, see *z/OS V1R4.0 and z/OS.e V1R4.0 Planning for Installation*, GA22-7504 for the z/OS release that you are installing.

## 4.9 z/OS installation using ServerPac

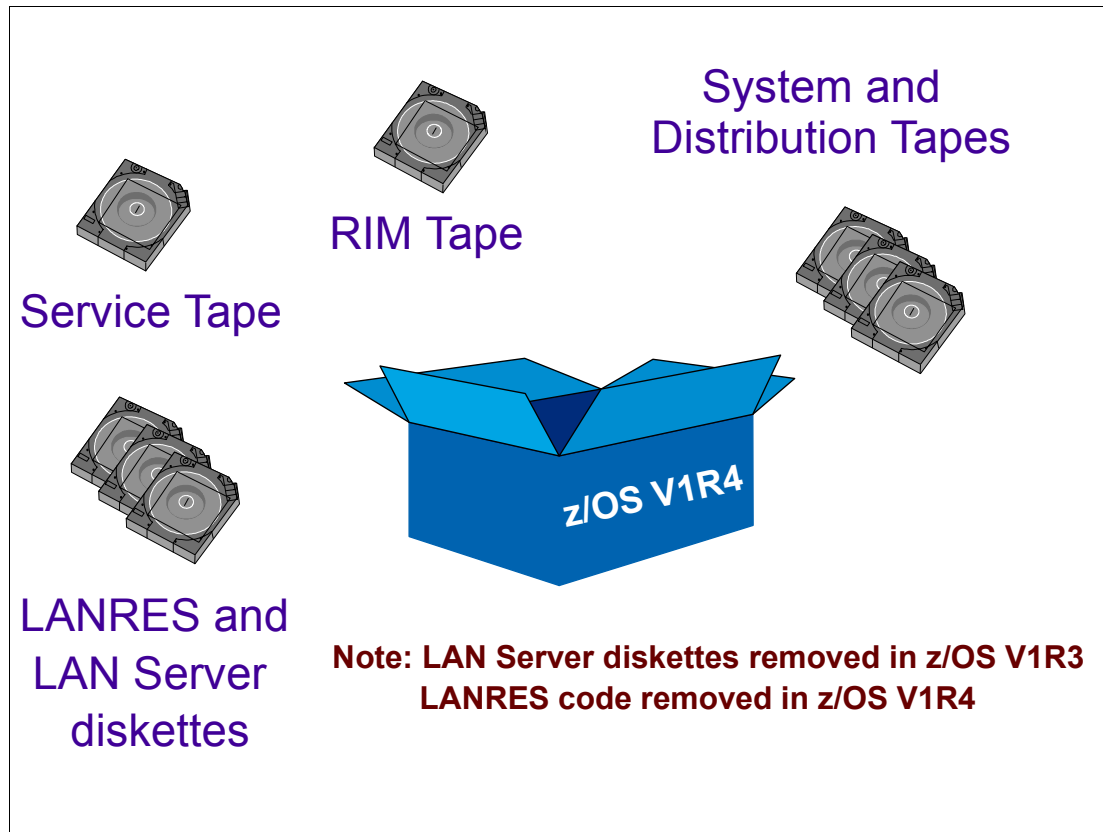


Figure 4-9 Installation tapes for ServerPac

Your z/OS ServerPac order contains an ISPF dialog that you use to install z/OS. This dialog is called the CustomPac installation dialog because it is used to install all of IBM's CustomPac offering.

Before you begin the installation, you should:

- ▶ Review the contents of the ServerPac shipment that you received from IBM by checking the packing slip to make sure that you have a complete set of installation tapes and documentation.
- ▶ Make sure your user ID has ALTER authority for the following high-level qualifiers:
  - CPAC
  - SYS1
  - All product-specific high-level qualifiers for products that come with your package. You can find a listing of all qualifiers by using the A-ALIAS option of the dialog, or refer to Product Information in the Appendix of the document *IBM ServerPac Using the Installation Dialog*, SA22-7815.
- ▶ During of the job phase of the installation process you may need RACF SPECIAL authority or equivalent if you use other security software.
- ▶ If you decided to use SMS to manage data sets in your order, your user ID needs READ access to FACILITY class, profile STGADMIN.IGG.DIRCAT.

## 4.10 Installing the CustomPac dialogs

- HLQ for the CustomPac data sets
- CustomPac data sets placement
- Extract LOADRIM
- Customize and run LOADRIM
- Start the CustomPac dialogs

```
//EXTRACT JOB <JOB statement info goes here...>
//*
//STEP01 EXEC PGM=IEBCOPY
//*
//SYSPRINT DD SYSOUT=*
//*
//IDOC DD DISP=SHR,DSN=SYS1.orderid.DOCLIB,LABEL=(06,SL),
// VOL=SER=tapeser,UNIT=tapeunit
//ODOC DD DSN=work.library.jcl,
// DISP=(NEW,CATLG,DELETE),
// VOL=SER=volser,UNIT=SYSALLDA,
// SPACE=(9600,(240,30,20))
//*
//SYSIN DD *
COPY INDD=IDOC,OUTDD=ODOC
S M=LOADRIM
/*
```

Figure 4-10 Installing the CustomPac dialogs

The first step in installing z/OS is to install the CustomPac dialogs from the RIM tape on your driving system. Once they are installed, the dialogs do not have to be reinstalled with every order. They are auto-upgraded whenever you get a new order. Version checking invokes the update of the dialogs during the CustomPac RECEIVE function.

Steps to install the CustomPac dialogs:

1. Define the HLQ for the CustomPac data sets (called *master* dialog data sets) pointing to a user catalog accessible by both the driving and the target systems; for example, SERVRPAC.  
Since the dialogs are permanently installed at your installation, you should not specify the IBM-supplied order number as the CustomPac qualifier.
2. If you intend to use SMS-managed dialog data sets, assign them to a management class that does allow migration, unless SMS and HSM environments will be shared between driving and target systems.
3. Unload the LOADRIM job from SYS1.orderid.DOCLIB data set the RIM tape, as shown in Figure 4-10.
4. The LOADRIM job contains steps to:
  - Delete previous CustomPac dialog data sets.
  - Unload master dialog data sets.
  - Allocate an order inventory data set to contain control information for all shipped ServerPac orders.



## 4.11 The RIM tape samples



Figure 4-11 Using the CustomPac dialogs

The RIM tape contains sample procedures, JCL, jobs, and CLISTs. They are in SYS1.orderid.DOCLIB. You can unload and modify these samples for your installation, as shown in Table 4-1.

Table 4-1 Useful samples from the RIM tape

Name	Description
LOADRIM	LOADRIM is the JCL to unload files from tape and set up the installation dialog. When you edit the LOADRIM sample JCL, you can choose the name of the master data sets, the unit name of your tape drives, and the VOLSER of the DASD which receives the installation dialog's data sets.
SETUP	This is a sample LOGON procedure which includes the CustomPac dialog ISPF libraries.
CPPCSAMP	This sample CLIST can be used to set up the environment instead of modifying the LOGON procedure. CPPCSAMP uses LIBDEFs and should be the preferred method to allocate the CustomPac libraries and start the dialog.
CPPINIT	With the CPPINIT CLIST, you can set up the environment from native TSO.
PRTDOC	This sample job prints the CustomPac Installation dialog reference manuals.

## 4.12 Starting the CustomPac dialogs

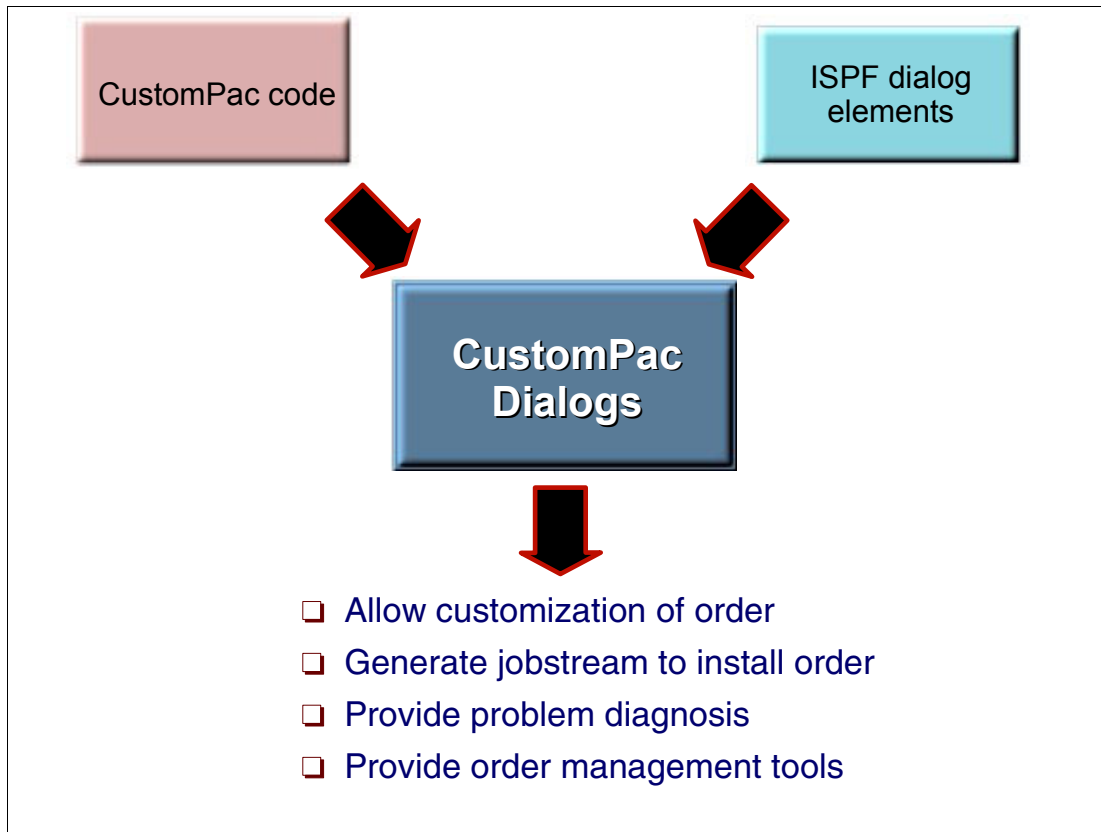


Figure 4-12 CustomPac dialogs

The CustomPac dialogs are now installed. To access the dialogs, we recommend that you:

1. Customize the CPPCSAMP CLIST, changing the *custompac.qualifier* according to the qualifiers of the master dialogs data sets.
2. Save the customized clist.
3. Go to TSO/ISPF, option 6 and execute the new CLIST:

```
exec 'DATA.SET.NAME(CPPCSAMP)'
```

Where DATA.SET.NAME is the partitioned data set where you saved the CLIST and CPPCSAMP is the member name where you saved the customized CPPCSAMP CLIST.

Figure 4-13 on page 175 shows the panel you receive when you start the CustomPac dialogs.

## 4.13 Receiving the ServerPac order

```
CustomPac ----- (C) IBM Corporation 1990-2003 -----
OPTION ==> R_

CustomPac INSTALLATION

    The HLQ of your MASTER Datasets is : SERVRPAC.MASTER

R   RECEIVE      - Receive an Order
I   INSTALL      - Install Orders
    ==>          (Order Number OR blank for ALL NEW Orders)
O   ORDERS       - Display Orders
```

Figure 4-13 CustomPac primary menu

Before you use the dialog to install your order:

- ▶ Review the recommended actions in “z/OS installation using ServerPac” on page 171.
- ▶ Be familiar with the dialogs. Consult *IBM ServerPac Using the Installation Dialog*, SA22-7815, in particular the following sections:
  - “Features of the Dialog Panels” describes the ISPF Edit settings used, the available primary and line commands, language setting, dynamic help, and diagnostic messages.
  - “Working With Your Order: An Overview of the Dialog Activities” summarizes the steps to install a ServerPac order.
  - “Using the Installation Menu” explains the dialog functions provided.

Once you start the dialog, the first thing to do is receive the order. This is done when you choose option **R** in the primary menu panel, as shown in Figure 4-13. Receiving the order means you copy the order from tape to DASD. This unloads the control tables and installation jobs from the shipment tapes to your DASD.

**Note:** HELP (PF1) is available on any panel. The HELP key is a very useful online help facility that explains every panel function in detail. Some panels have PRIM and LINE commands available. Using the HELP key allows you to get a description and example of how to use the commands.

## 4.14 Order Receive panel

```
CustomPac ----- Order Receive -----
COMMAND ==>

ORDER DETAILS

    Order Number  ==> OS261466
    TAPE VolSer   ==> R1466A      TAPE Unit   ==> 3490
    Order HLQ     ==> SYSP0K.OS261466
    DASD VolSer   ==> TOTTS2      DASD Unit   ==> SYSDA

-----
Do You want to Use          Edit JOB Stream
VB Clists                   ==> N      Before Submitting    ==> Y

Is this Order a SystemPac
or a ServerPac              ==> N
-----
```

Figure 4-14 Order Receive

Since you chose option **R**, you receive the panel shown in Figure 4-14. After completion of the Order Receive panel details, the JCL generated is shown to you and submitted to download the order installation libraries from the shipment tape to DASD.

The following information is entered on the Order Receive panel:

- Order Number** Your specific IBM-supplied order number, which is listed on the cover of the order documentation.
- TAPE VolSer** The volume serial number of the RIM tape.
- TAPE Unit** The unit type of your tape drives.
- Order HLQ** The HLQ used to allocate the order installation data sets. We recommend you include the order number as part of the qualifier.
- DASD VolSer** The VOLSER of the DASD where your order data sets are to be restored.
- DASD Unit** The unit type of your DASD units and is defaulted to SYSDA.

Press Enter after you have filled out the order details. The Generate Jobstream panel appears.

## 4.15 Generate Jobstream panel

```
CustomPac ----- Order Receive -----
COMMAND ==>

GENERATE JOBSTREAM

  Enter JOBCARDs

> //ZOSOREC   JOB (XXXX), 'PROGRAMMER NAME',           <
> //          CLASS=A,MSGCLASS=X,                     <
> //          NOTIFY=&SYSUID,REGION=4M                 <
> //*                                                  <

Installation  ISPLLIB ==> ISP.SISPEXEC
              ==>
           ISPF   ISPMLIB ==> ISP.SISPMENU
              ==>
Libraries     ISPPLIB ==> ISP.SISPPENU
              ==>
              ISPSLIB ==> ISP.SISPSENU
              ==> ISP.SISPSLIB
              ISPTLIB ==> ISP.SISPTENU
              ==>
```

Figure 4-15 Generate Jobstream panel

In this panel, enter the job card information relating to your installation standards. Change the ISPF library names to correspond to your current ISPF environment.

After pressing Enter, you receive a panel where you can specify additional job card information for loading of the RIMs. Depending on whether you previously indicated that you wanted to edit the job stream before submission, you can now review and edit the generated job that is to receive the order, then submit it.

After successful completion of the job, your order data sets are copied from the RIM tape to your DASD.

When you selected an order for processing, an enqueue is issued against the order number. This ensures that only one person can work on an individual order at any one time.

## 4.16 Selecting an order to Install

```
CustomPac ----- ORDER INSTALLATION ----- Row 1 to 1 of 1
COMMAND ==>                                     SCROLL ==> PAGE

PRIM Cnds:(? SET L F N P REFresh SStatus SORT VERbose VERsion)
LINE Cnds:<Copy Delete Edit Finalise Insert Products Report Select>

  ORDer    PROFile    SYStem      PAC      Prod      Change    Change
S  Number  Number      Name      SREL  TYPE      DATE      St  USER      DATE
-  - - - -  - - - - -  - - - - -  - - - -  - - - - -  - - - - -  - - - - -  - - - - -
S OS261466                OS390R6  Z038  EXP      01/08/1996 S  FHOFMA  20/08/1996
***** Bottom of data *****
```

Figure 4-16 Order Installation panel

After you finish the order receive function, place an I on the CustomPac Installation Panel to start the installation of the order. Then you receive the panel shown in Figure 4-16.

On the Order Installation Panel, you can now select the order you want to install. If this is your first ServerPac installation, only one order number can be selected.

## 4.17 Installation dialog

```
CPPPFLOW ----- (C) IBM Corporation 1990-2002 -----
OPTION ==> C

Installation

Order ( 0S261466 ) The Following Functions MUST be Executed in Sequence

  C   Configure      Select Configuration for Installation and Merge
  *   Variables      Define Installation Variables
  *   Zones          Define Zone Configuration
  *   Modify         Modify System Layout
  *   Alias          Define Alias to Catalog Relationships
  *   SSA            Define SSA to Catalog Relationships
  *   Installation   Select and Submit Installation Jobs
  *   Save           Save Used Configuration
  *   Update         Update Order Inventory Status
  DI  Display        Display a List of Dataset Names
  DT  Display        Display a Summary of Order Tables
```

Figure 4-17 Installation dialog

After selecting a ServerPac order to install, the main installation dialog panel is invoked. When this panel is shown for the first time during a ServerPac order installation, the only options available are:

- C** Option C on this panel allows you to select a configuration for merging an initial installation. If this is your first CustomPac installation, the Create Configuration panel appears.
- DI** Option DI allows you to display any data set names and is similar to the PDF Option 3.4 function.
- DT** This option displays summary information about your order and your work configuration, such as order variables, zone nicknames, logical volumes, alias to catalog mapping, catalog to SSA mapping, and installation jobs.

The other options are now marked with an asterisk (\*) and become available as long as the previous function has successfully finished. Now, choose option **C**.

## 4.18 Choosing the installation type

```
CPPP6015 ----- CREATE Configuration (0S261466) -----
OPTION ==> F

Select the Install type :

F - Full System Replacement installs a complete new IPL-able
standalone system including all SMP/E-maintained libraries, SMP/E
environment, operational data sets, and CustomPac sample data sets.
The supplied operational data sets must be merged with or replaced
by production operational data sets before the new system is used
in production.

S - Software Upgrade installs only the SMP/E-maintained libraries,
SMP/E zones, and CustomPac sample data sets. Operational data sets,
including system control files (like LOGREC and VTAMLST), a security
system database, and a master catalog must already exist. These
existing operational data sets must be updated as required for new
products and product changes before the first IPL.

For more information about Software Upgrade, enter ? in the option field

.
```

Figure 4-18 Create Configuration: Choosing the Installation Type

When you use the dialog to select an order, its shipped configuration is added to the dialog. You must select and create a configuration to start the installation. Option **F** installs a complete z/OS system. It installs all data sets needed to IPL, log on to the target system, and run a z/OS image to complete other installation and customization tasks, including:

1. System software and related data sets, like distribution, target, SMP/E libraries, and so forth.
2. System data sets like page data sets, system control files, and master catalog.

Because IBM creates a working set of operational data sets for you, a full system replacement helps assure a successful first IPL.

Choose Full System Replacement, Option **F**.



## 4.19 Selecting a JES for the configuration

```
CPPP6016 ----- JES Element Selection ( OS261466 ) -----  
COMMAND ==>  
  
Choose JES elements to be installed:  
  
          JES Elements to Install ==> JES2      (JES2, JES3, or BOTH)  
  
Specify options for merging JES SMP/E target and DLIB zones:  
  
Merge JES2 SMP/E Zones into BCP Zones ==> Y  (Y or N)  
Merge JES3 SMP/E Zones into BCP Zones ==>    (Y or N)  
  
Note: If you wish to merge a JES element's zones, it must have been  
selected for installation above. For more information, enter ? in the  
Command field.
```

Figure 4-19 JES Element Selection panel

Select at least one JES element for installation: JES2 (and SDSF) or JES3. If your installation requires both JES elements to be installed, you can select BOTH.

For each JES element you select, specify whether the dialog is to merge the SMP/E zones of the JES element with the Base Control Program (BCP) zones. When you merge a JES element with the BCP zone, no separate zone is created for the JES element.

If you plan to migrate to subsequent releases of JES2 (and SDSF) or JES3 when you migrate to the next level of z/OS, we recommend that you specify Y on this panel to merge the selected JES elements into the BCP zone.

However, if you plan to stage your z/OS and JES migrations separately, do not merge zones.

Once the panel information is filled out, press Enter. The Create Configuration panel is displayed.

## 4.20 Create Configuration panel

```
CPPP6011 ----- CREATE Configuration (0S261466) ----- Row 1 to 2 of 2
COMMAND ==> CR                                     SCROLL ==> CSR

Select Configuration

PRIM Cnds:(? SET L F N P SORT CReate)
LINE Cnds:(Select)

S CONFIguration                                     Comment
-----
* SYSP0K.0S261466                                  Always Selected for Order
-----
* SYSP0K.MASTER|                                   MASTER Configuration
-----

**NO MERGABLE ORDER CONFIGURATIONS|

***** Bottom of data *****
```

Figure 4-20 Create Configuration panel

Before you start the installation, you must select and create a configuration. On the Create Configuration panel, you can see the master configuration and, if available, other saved configurations. The shipped configuration is always automatically selected.

Use the **CR** (CREATE) command to create a work configuration. If you are using the dialog for the first time, begin by using the configuration that IBM supplies. Later in the installation, you will be able to save this configuration for use with subsequent ServerPac orders. If you are merging this order with a saved configuration, see “Merging a Configuration with a Previous Order” in *IBM ServerPac Using the Installation Dialog*, SA22-7815, for further considerations.

Enter **CR** in the command line to create a work configuration *or* type an **S** in front of the configurations you want to merge, if applicable. Press Enter.

The dialog displays the Configuration Profile panel.

## 4.21 Configuration Profile panel

```
CPPP6014 ----- CREATE Configuration ( 0S261466 ) ROW 2 TO 14 OF 15
COMMAND ==>                                     SCROLL ==> HALF

Configuration Profile

PRIM Cnds:(? F N P SAVE)

-----
*****
* SAVE CONFIGURATION *
*****

TABLE(NEWA2CD) SAVED - ALIAS TO CATALOG RELATIONSHIPS
TABLE(NEWCD2S) SAVED - CATALOG TO SMS RELATIONSHIPS
TABLE(NEWDEVT) SAVED - DEVICE TYPES
TABLE(NEWFTBL) SAVED - INSTALLATION FLOW
TABLE(NEWITBL) SAVED - INSTALLATION JOBS
TABLE(NEWLTBL) SAVED - LOGICAL VOLUME DATA
TABLE(NEWPTBL) SAVED - PHYSICAL VOLUMES
TABLE(NEWHTBL) SAVED - MERGE DATA SETS
TABLE(NEWVTBL) SAVED - INSTALLATION VARIABLES
TABLE(NEWZTBL) SAVED - ZONE CONFIGURATION TABLE

***** BOTTOM OF DATA *****
```

Figure 4-21 Configuration Profile panel

The Configuration Profile panel shows which variables and jobs have been merged, if any, and the tables that have been saved. In the profile, the following mnemonics identify source data *for the merge*:

- ▶ CPPSENUC: Saved configuration skeleton library
- ▶ CPPTENUC: Saved configuration table library
- ▶ CPPSENUM: Master skeleton library
- ▶ CPPTENUM: Master table library

Enter **SAVE** (or press the End key) to save the configuration profile. The Create Configuration panel is displayed again, with the message "Work Configuration Created." Press Enter to return to the Installation Menu.

Select **V**, Define Installation Variables, from the Installation panel.

## 4.22 Installation Variables panel

```

CPPP6111 ----- Installation Variables ( OS261466 ) - Row 1 to 26 of 62
COMMAND ==>                                     SCROLL ==> PAGE

Variable Selection List                          SHOW( *      )

PRIM Cnds:(? SET L F N P CANCEL SAVE SHOW VARname)
LINE Cnds:(Browse Delete Edit Insert Repeat Ship)

S   Synonym          STA Contents
-   -----          -   -+--+0--+--+0--+--+0--+--+0-----
==> GEN SYSTEMPAC
    UNIT SYSDA         D SYSDA
    TGT UNIT TYPE      D
    TARGET VOLSER      D
    ASSEMBLER NAME     D ASMA90
    SMPTLIB PREFIX     D SYS1.MVS
    SMPTLIB UNIT       D 3390
    SMPTLIB VOLSER     D MVSCAT

==> HFS INFORMATION
    INSTALL DIRECTORY  D /Service

==> MVS NEW ADR
    ADR NEW MCON       D 0F2

==> MVS NEW DSN
    TIMEZONE GMT       P W.05.00.00
    SYSNAME            D CPAC
    RACF OLD PDSN     D DRIVSYS.RACFP.DSN
    RACF OLD BDSN     D DRIVSYS.RACFB.DSN
    RACF OLD PVOL     D MVSRES
    RACF OLD BVOL     D MVSRES

==> MVS NEW VOL
    TARGET RACF VOLSE  D MVSCAT

```

Figure 4-22 Installation Variables panel

The Installation Variables panel is now displayed.

The Variable Selection List displays variables that are required to install your order. The list reflects the installation type you chose (full system replacement or software upgrade). Use this panel to set the values of these variables appropriately for your environment. Refer to the “Variables” appendix in the *ServerPac: Installing Your Order* publication that comes with your ServerPac, for the exact variables for your order along with a brief description of each.

Verify the current contents and enter or change any values by overtyping in the Contents column if a value is either missing or invalid. You cannot update variables with a status of C (customized).

It is recommended that you read “Defining Installation Variables” in *IBM ServerPac Using the Installation Dialog*, SA22-7815, before changing any installation variable values.

The variable for AUTH.LINKLIB may be an existing authorized library of your installation site. You can use the **VAREdit** command on some panels to change the installation variables later.

Once you have customized the installation variables, return to the Installation panel and choose function **Z**: Define Zone Configuration.

## 4.23 Define ZONE Information panel

```
CPPP6391 ----- Define ZONE Information ( 10S261466 ) --- Row 1 to 5 of 5
COMMAND ==>                                     SCROLL ==> HALF
PRIM Cnds: (? SET L F N P SORT CANce1 SAVE)
LINE Cnds: (feaTures)

All DLIB Zones will be allocated in CSI : SMPE.MVS.DLIB.CSI
All TLIB Zones will be allocated in CSI : SMPE.MVS.TARGET.CSI

      S NickName      DLIB Zone      Target Zone      SST
      - - - - -      - - - - -      - - - - -      - - -
          100          MVSD100          MVST100          MVS
          200          MVSD200          MVST200          MVS
          300          MVSD300          MVST300          MVS
          400          MVSD400          MVST400          MVS
          500          MVSD500          MVST500          MVS
***** Bottom of data *****
```

Figure 4-23 Define ZONE Information panel

This brings you to the Define ZONE Information panel, where you can define your SMP/E zone configuration. This panel is displayed even if you do not plan to change the shipped zone names.

You can change the zone names to the names you want. The nickname is used to pair them together.

The reason for having more than one target and DLIB zone is that you cannot have incompatible products together in one SMP/E zone, such as COBOL/II and z/OS LE.

Use the **SHIP** command with caution, because it restores all DLIB and target zone names to their shipped value.

For more information, refer to *ServerPac: Installing Your Order*.

After your changes, if any, return to the Installation panel and enter **M** to begin the next dialog function.

## 4.24 Modify System Layout Options panel

```
CPPPP605T ----- Modify System Layout Options ( I0S261466 ) -----
OPTION ==>

A   Create a Recommended System Layout (Automatically assign target and
    DLIB data sets to physical volumes by data set type)

C   View and change data sets by selected attributes
T   View and change device type table (DEVT)

D   Data Set Summary (SUMD)
M   Merged Data Set Summary (SUMD M)
S   Shipped and Merged Data Set Summary (SUMD S)
U   User Data Set Summary (SUMD U)

V   Physical Volume Summary (SUMP)
L   Logical Volume Summary (SUML)

P   Product, Feature, and Element Summary

----- Session Control Options -----
K   Keep Changes made in this dialog session so far (SAVE)
B   Back Out changes from this dialog session (CANCEL)
```

Figure 4-24 Modify System Layout Options panel

Defining the target system layout is one of the most important steps during order installation. During this part of the dialog, you create the data set layout for your new system. After you have modified this configuration, you can save it for merging with future ServerPac installations.

You can create the new data set layout in one of three ways:

- ▶ Option **A**: Recommended System Layout to make the dialog automatically assign the target and DLIB data sets in the configuration to physical volumes. The dialog does not automatically assign any SMS-managed data sets in the configuration.
- ▶ Option **C**, View and Change, to assign your order's data sets to volumes by displaying groups of data sets, and using the **CHANGE PVOL** command to specify their placement on physical volumes.
- ▶ Other options (D, M, S, U, V, L, and P) to assign your order's data sets to logical volumes and then assign those logical volumes to physical volumes (DASD).

The recommended system layout provides a foundation for the ongoing growth and maintenance of your system. When you group your system's data sets by their content and importance to your installation, you help to minimize the complexity of future installations.

Read and use the section "Modifying the System Layout" in *IBM ServerPac Using the Installation Dialog*, SA22-7815. The *ServerPac: Installing Your Order* publication that comes with your order also contains all information relating to the products to be installed.

## 4.25 Summary of features and elements

```
CPPP6051 ----- Modify System Layout ( OS261466 ) - ROW 1 TO 8 OF 16
COMMAND ==>                                     SCROLL ==> PAGE

SUMMARY Of Features/Elements

PRIM Cnds:(? SET F L N P SORT CANce1 SAVE DEVT SUMD SUML SUMP)
LINE Cnds:<Dslist Select>

  S  Feature/Element                                Data sets
  -  -----
  BCP                                           70 ( 6)
  BDT                                           9 ( 0)
  BDT FILE-TO-FILE                               8 ( 0)
  BM BOOKSERVER      2.01.0                       9 ( 1)
  BOOKMANAGER BUILD  ENU                          44 ( 4)
  BOOKMANAGER READ  ENU                           16 ( 3)
  C/C++                                          61 ( 1)
  CUSTOMPAC JES2 SYSTEM DATASETS                   3
  CUSTOMPAC JES3 SYSTEM DATASETS                   7
  CUSTOMPAC OPERATIONAL AND SAMPLE DATASETS        29
  CUSTOMPAC SMP/E DATASETS                          30
```

Figure 4-25 Summary Of Features/Elements panel

When you select Option **P** of the Modify System Layout Options panel, the Summary of Features/Elements panel shown in Figure 4-25 is displayed.

This panel allows you to manually customize individual data sets, logical volumes, and physical volumes. The panel lists the products, features, and elements shipped in your ServerPac order, and the following CustomPac-specific data set groups:

- ▶ CustomPac SREL-specific SMP/E data sets
- ▶ CustomPac operational and sample data sets
- ▶ CustomPac JES2 data sets
- ▶ CustomPac JES3 data sets
- ▶ CustomPac SMP/E data sets

The panel shows primary and line commands you can issue to perform the actions you need to customize volumes and data sets names. For example, the **Select** command entered next to a product displays the Logical Volume By FEATURE/ELEMENT panel for the selected product, where you can use the line command **Assign** to assign all data set profiles for the selected logical volume to a different logical volume.

The **D** line command returns the Summary of Data Sets panel shown in Figure 4-26 on page 188.

## 4.26 Summary of data sets of a feature/element

```
CPPP6052 ----- Modify System Layout (0S261466 ) - ROW 1 TO 10 OF 10
COMMAND ==>                                     SCROLL ==> PAGE

SUMMARY OF DATA SETS

PRIM Cnds:(? SET L F N P SORT Change OFile OList FC VARIable VERbose)
LINE Cnds:(Merge eXpand Conflict Unmerge Attribs Space Resolve)

S  DSName                                X F RECFM DSORG LRECL BLKSZ  R
-  -----                                - - - - - - - - - - - - - - -
ISP.AISPALIB                             *  FB  PO      80 23440  Y
ISP.AISPEXEC                             *  FB  PO      80 23440  Y
ISP.AISPGENU                              *  FB  PO      80 23440  Y
ISP.AISPMACS                             *  FB  PO      80 23440  Y
ISP.AISPMENU                              *  FB  PO      80 23440  Y
ISP.AISPMOD1                              *  U  PO       0 32760  N
ISP.AISPPENU                              *  FB  PO      80 23440  Y
ISP.AISPPUBS                              *  FB  PO     4096 16384  Y
ISP.AISPSLIB                              *  FB  PO      80 23440  Y
ISP.AISPTENU                              *  FB  PO      80 23440  Y
***** BOTTOM OF DATA *****
```

Figure 4-26 Displaying Data Sets of a Feature/Element

Use the Summary of Data Sets panel to do any of the following:

- ▶ Merge or unmerge ServerPac-shipped data sets (you cannot merge or unmerge user-defined data sets).
- ▶ Modify the attributes of particular data sets or modify their space information.
- ▶ Make global changes to multiple data sets.
- ▶ Write a list of the data sets in the ISPF LIST data set or a user-defined file.

The **Change** primary command allows you to make global changes to data set profiles. For example, you can change the HLQ for those product data sets. The line commands **A** and **S** allow you to change data set name, logical volumes space, and BLKSIZE definitions for a specific data set profile.



## 4.27 Summary of Physical Volumes panel

```

CPPP605K ----- Modify System Layout (0S261466 ) --- ROW 1 TO 8 OF 8
COMMAND ==>                                     SCROLL ==> PAGE

SUMMARY Of Physical Volumes

Primary Commands: (? SET L F N P SORT DEVT)
Line Commands: <Assign Dslist>

  PVolume/  Seq.      Device  Warn-  Exist  ----- Cylinders -----
S  STORCLAS No.  CCUU  Type   ings  Data  RSVD  Shipped Used  Free
-  - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
  MVSCAT          04C7  3390-3  OVR<<S  Y      0     2706      0    633
  MVSDL1         D01  CCUU  3390-3  OVR<<S  N      0     2939      0    400
  MVSDL2         D02  CCUU  3390-3  OVR<<S  N      0     1323      0   2016
  MVSRS1         T01  CCUU  3390-3          N      0     2871      0    468
  MVSRS2         T02  CCUU  3390-3          N      0     2065      0   1274
  MVSRS3         T03  CCUU  3390-3          N      0     2724      0    615
  SMSDCLAS
  SMSRCLAS
                                     1115
                                     2157
***** BOTTOM OF DATA *****

-----
| CPP0605059S Volume (MVSCAT) is overallocated. |
-----

```

Figure 4-27 Summary of Physical Volumes

Before you leave the Modify System Layout main panel, you should enter the **SUMP PRIM** command, which displays a summary of the physical volumes.

When one of your physical volumes becomes over-allocated, the following message appears on the panel:

```
| CPP0605005S At least ONE PHYSICAL Volume is OVER ALLOCATED |
```

This condition is also shown by the <<<<<< next to the physical volume names.

By using the dialogs previously described, you are able to modify the system layout and correct the over-allocation of physical volumes.

**Important:** Use the **SHIP** command with care because it is powerful. This command is available on several dialog panels. It can be used to restore all profiles to their initial-ship values. You can lose all the customization you previously entered if you issue the **SHIP** command.

## 4.28 Creating the recommended system layout

```
CPPP625B ----- Automatic Data Set Assignment ( 03261466 ) -----
OPTION ==>

    A - ALL          Assign all target and DLIB data sets in the configuration
                    to physical volumes automatically.  This option creates a
                    recommended system layout.

    N - NEW          Add new data sets to an existing configuration.  This
                    option automatically assigns all new data sets, but
                    preserves the placement of previously-assigned data
                    sets in your saved configuration.

    P - PARTIAL      Assign new data sets and reassign some existing data sets
                    to physical volumes.  This option automatically assigns
                    all new data sets to physical volumes, as well as data
                    sets from selected volumes in the saved configuration.

Default Device Type ==> 3390-3 (For example, 3390-3)
```

Figure 4-28 Automatic Data Set Assignment Panel

When you select option A, the Recommended System Layout, the panel shown in Figure 4-28 is displayed.

The dialog automatically assigns some or all of the target and DLIB data sets in your order to DASD volumes based on the following considerations:

- ▶ Whether the data set is a target data set or a DLIB data set
- ▶ Whether the data set must reside on the IPL volume
- ▶ The type of data in the data set: panels, messages, load modules, and so on
- ▶ If the data set should reside on a particular volume in the configuration; for example, the first volume or one of the last volumes

The dialog does not automatically assign an order's operational data sets or any of the sample CustomPac data sets. You must place these data sets yourself.

The Default Device Type field specifies the type of device to be used if the dialog creates more volumes for data set assignments. Enter ? in the Default Device Type field and press Enter to see a list of other available devices. A configuration can include more than one device type.

Choose **ALL** when you are installing a ServerPac order for the first time, or if you are not using a saved configuration as the basis of your new system. This approach creates a new configuration based only on the new order to be installed.

## 4.29 Current Volume Configuration panel

```
CPPP625C ---- Automatic Data Set Assignment ( 0S261466 ) ---- ROW 1 TO 7 OF 7
COMMAND ==> SCROLL ==> PAGE

Current Volume Configuration Scope==> ALL

Primary Commands: (? Reset Create)
Line Commands: (Select Insert List Move After Before eXclude)

  Phys.  Volume  Sequence  Device  Reserved  Used +  Volume  Existing
  S Volume Type   Number   Type     Space   Reserved Threshold Data
  - - - - -
  MVSRES TARGET T01     3390-3   N        315 %   85%     N
  MVSDLB DLIB    D01     3390-3   N        224 %   85%     N
***** Bottom of Data *****
```

Figure 4-29 Automatic Data Set Assignment: Current Volume Configuration panel

After you select a setting for automatic assignment, the dialog displays the current volume configuration, as shown in Figure 4-29.

As shipped by IBM, a new configuration consists of a target volume, MVSRES; a DLIB volume, MVSDLB; and a catalog volume, MVSCAT. Because MVSCAT contains only operational data sets, this volume is excluded from automatic assignments, and therefore is not shown in the panel display.

In this panel you use line commands in the *S* column for the following purposes:

- ▶ **S** to select a volume to change
- ▶ **L** to list data sets currently assigned to a volume
- ▶ **I** to insert more volumes
- ▶ **X** to make exceptions to volume assignments. This command has a different use according to the scope of automatic assignment (ALL, NEW, or PARTIAL) selected on the panel shown in Figure 4-28.
- ▶ **M**, **A** or **B** to move volumes in the order in which you want them to be used.

If you plan to rename these volumes, select the volumes now through line command **S** and rename them as needed. Later, when the new configuration is created, it is more difficult to rename these volumes. When you select **S** the panel shown in Figure 4-30 on page 192 is shown.

## 4.30 Display and Change Volume Attributes panel

```
CPPP625D ---- Automatic Assignment - Attributes ( OS261466 ) -----  
COMMAND ==>  
  
Display and Change Volume Attributes  
  
Volume Serial ==> MVSRS1  
  
Device Type ==> 3390-3 (For example, 3390-3)  
Volume Type : TARGET (Target, DLIB or Both)  
  
Reserved Space ==> 0 (In Cylinders)  
  
Existing Data ==> N (Y or N)
```

Figure 4-30 Display and Change Volume Attributes

*Changing the Existing Data field from YES to NO causes the volume to be initialized by the installation jobs, and any existing data on the volume is lost.*

To save your changes to the volume, press the Enter key and you return to the Current Volume Configuration display.

Once you have customized the system layout, return to the Installation panel for the next function.

## 4.31 Define alias-to-catalog relationships

```

CPPP6021 ----- ALIAS to CATALOG ( OS261466 ) ---- ROW 1 TO 9 OF 9
COMMAND ==>                                     SCROLL ==> PAGE

Define CATALOG Data set Names

PRIM Cnds:(? SET L F N P SORT CAnce1 SAVE)
LINE Cnds:(Delete Insert Repeat)

  S  Alias      STA TARGET System Catalog DSName                Type
  -  - - - - -  - - - - - - - - - - - - - - - - - - - - - - - - -
COB2          ????????.CATALOG
GIM           ????????.CATALOG
ICQ           ????????.CATALOG
CPAC          M ?MASTER.CATALOG                                MCAT
ISP           ????????.CATALOG
ISR           ????????.CATALOG
NETVIEW       ????????.CATALOG
SMPE          ????????.CATALOG
SYS1          M ?MASTER.CATALOG                                MCAT
***** BOTTOM OF DATA *****

```

Figure 4-31 Define CATALOG Data set Names

From the Installation Menu, enter A, Define Alias to Catalog Relationships.

In this option, you specify the catalog data set name for each *alias*. The dialog and the installation process for ServerPac use the standard order of catalog search when defining and locating data sets. Therefore, there must be an alias in the target system master catalog for each high-level qualifier used for data sets that are to be cataloged in a user catalog. Also, there must be aliases in the driving system master catalog for the system-specific aliases (SSAs) you chose to use when installing the order.

Before you begin to define the alias-to-catalog relationships and system-specific alias-to-catalog relationships, you should read Chapters 9 and 10 in *IBM ServerPac Using the Installation Dialog*, SA22-7815, to become familiar with using system specific-aliases and the catalog structure. Also, those chapters contain worksheets models you can use to plan:

- ▶ The catalog and alias names, and their relationships
- ▶ The catalog names and their associated SSAs

Use the Alias to Catalog panel to specify which HLQ you want to be associated with a catalog. An *M* in the STA column indicates that this alias name must be associated with a master catalog. The ?????? in the TARGET System Catalog DSName field indicates that there is no catalog defined yet. This function allows you also to insert additional user-defined alias names and catalogs.

After specifying the alias-to-catalog relationship, you can select *SSA* on the Installation panel, which leads you to the SSA-to-Catalog panel shown in Figure 4-32 on page 194.

## 4.32 Define system-specific alias (SSA)

```

CPPP6031 ----- SSA to CATALOG ( 0S261466 ) ----- ROW 1 TO 4 OF 4
COMMAND ==>

CATALOG Selection List

PRIM Cnds:(? SET L F N P SORT CANcel SAVE)
LINE Cnds:<Select>

                                     Allocate  Define
S Catalog Name                      SSA Name Type VOLUME | | Unit
-----
TONY.CATALOG                        ?      UCAT AMHCAT Y  Y 3390
NET.CATALOG                         ?      UCAT NETCAT Y  Y 3390
SMOMP.CATALOG                       ?      UCAT SMCAT  Y  Y 3390
TSO.CATALOG                         ?      UCAT TSOCAT Y  Y 3390
VS4.MASTCAT                         ?MCAT  MCAT MVSCAT Y  Y 3390
***** BOTTOM OF DATA *****

```

Figure 4-32 Defining system-specific aliases

Many of the data sets in your new order already exist on your driving system. While your order's data sets are intended for creating a new target system, there is a period during installation in which jobs running on your driving system must be able to locate the target system's data sets.

Because many of these data set names already exist in your master catalog, ServerPac requires a way to find the data sets in the normal order of catalog search. This way, jobs on the driving system can locate the target system's data sets without disturbing the operation of the driving system. Without such a method, ServerPac could not build a target system with any data sets that were already cataloged and allocated on your driving system.

In the Define SSAs function of the dialog, you define temporary high-level qualifiers (HLQs) for the target system data sets. During the installation, your order's data sets are cataloged with the temporary HLQs. To direct the catalog entries to the proper catalog, the temporary HLQs are defined as aliases in the driving system's master catalog. Thus, these alternate HLQs are called system-specific aliases or SSAs. Later, during the installation, jobs rename the target system's data sets to their true names, and an optional job is provided for you to remove the SSAs.

The SSAs you specify here are used to create alias entries for these catalogs in the driving system's master catalog. The process you use to define SSAs depends on your installation type: full system replacement or software upgrade. For a full system replacement, define your SSA and catalog with the panel shown in Figure 4-33 on page 195.

## 4.33 Define SSA and CATALOG Data panel

```
CPPP6033 ----- SSA to CATALOG (0S261466 ) -----
COMMAND ==> _

Define SSA and CATALOG Data

    Catalog : TONY.CATALOG
    Type    : UCAT

    Define SSA      ==> Y    ( Y or N )
    Allocate Catalog ==> Y    ( Y or N )

    SSA Name       ==> SMPESSA

    Catalog Volume  ==> AMHCAT  (? for List of Available Volumes)

    If allocating the catalog, the following information is required:

    Primary Space   ==> 12      (1-999 Cylinders)
    Secondary Space ==> 12      (1-999 Cylinders)
```

Figure 4-33 Define SSA and CATALOG Data

The panel shown in Figure 4-33 is displayed when you use full system replacement, option **F** on the panel shown in Figure 4-18 on page 180.

The panel fields are as follows:

- ▶ **Catalog:** Name of the catalog for which an SSA is to be defined.
- ▶ **Type:** The catalog type. MCAT indicates a master catalog; UCAT indicates a user catalog.

Define the following fields:

- ▶ **Define SSA:** Set this value to Y (yes) to define a new SSA in the driving system's master catalog. Set this value to N (no) if the SSA is already defined in the driving system's master catalog. If you set the Allocate Catalog field to Y, you must set the Define SSA field to Y.
- ▶ **Allocate Catalog:** Specifies whether the catalog does not yet exist on the target system, and is to be physically allocated (Y); or N if the catalog already exists on the target system.
- ▶ **Catalog Volume, Primary Space and Secondary Space:** Specifies allocation parameter for the catalog.

This is the end of the customization steps for the ServerPac. You are now ready to run the supplied installation jobs. From the Installation Menu, enter **I** and the panel in Figure 4-34 on page 196 is shown.

## 4.34 Job Selection List panel

```
CPPP6121 ----- Installation JOBS (0S261466 | ----- Row 1 of 159 -----
COMMAND ==> SCROLL ==> HALF

JOB Selection List SS$( EXCLUDE )

PRIM Cnds:(? SET L F N P GENskel Ofile OList SUMmary SS$ VARedit)
LINE Cnds:(Backup Delete Edit Insert Log Output Select SS-block Vars)

S      Description                                STEP      MC STATUS      RC
-----
SRC DEFAULT JOBCARD

==> INSTALLATION JOBS
DOC RUNNING INSTALLATION JOBS
DOC DIALOG VERIFY
DOC INSTALLATION SETUP JOBS
JOB INITIALIZE REQUIRED DASD                OFFLINIT 00
JOB UNLOAD DOCLIB FROM TAPE TO DASD        UNLODOC  00
JOB UNLOAD SCPPLENU FROM TAPE TO DASD      UNLDSCPP 00
JOB UNLOAD INSTGUID FROM TAPE TO DASD      UNLDBOOK 00
DOC DEFINE CATALOGS AND RESTORE
JOB RACF PROFILES ON DRIVING SYSTEM        RACFDRV  00
JOB DEFINE CATALOGS                        DEFCAT   00
JOB DEFINE SYSTEM-SPECIFIC ALIASES         DEFSSA   00
JOB ALLOCATE AND CATALOG DS                ALLOCDS  00
```

Figure 4-34 Job Selection List

There are three types of components shown on the Installation Jobs panel:

<b>SRC</b>	Source data such as parameter lists
<b>DOC</b>	Documentation
<b>JOB</b>	Executable JCL

The installation steps are grouped into the following sections:

- ▶ Package-specific installation
- ▶ Product-specific installation
- ▶ Post-installation
- ▶ Additional post-installation
- ▶ Customization section
- ▶ Installation verification section
- ▶ Cleanup jobs
- ▶ Migration section
- ▶ Customer-specific customization

When you enter the Installation Jobs panel for the first time, the installation jobs have still not been generated. All installation jobs are generated using ISPF tailoring services. We recommend that you use the **GENSKEL** command to tailor all of the installation jobs at one time. When GENSKEL completes, the dialog saves the jobs in a back-up data set.

For more information about the commands you can use on this panel, refer to *IBM ServerPac Using the Installation Dialog*, SA22-7815, chapter 11.



## 4.35 GENERATE File Tailored Installation Jobs

```
CPPP6126 ----- Installation JOBS ( 0S261466 ) -----  
COMMAND ==>  
  
GENERATE File Tailored Installation Jobs  
  
This function generates a BATCH job which will file tailor  
ALL Installation Jobs in one pass, and save the jobs to the  
BACKUP dataset.  
  
If a job already exists in the backup dataset  
  
REPLACE Job ==> Y (Y or N)  
  
Note: After submitting the GENSKEL job, you must exit  
the dialog to release GENSKEL processing. Also,  
to avoid dataset contention, you may not invoke  
the dialogs until the GENSKEL job has completed.
```

Figure 4-35 GENERATE File Tailored Installation Jobs Panel

When you enter the **GENSKEL** command, the panel shown in Figure 4-35 is displayed.

The **GENSKEL** command submits a batch job, which generates all the installation jobs. Each job is stored in the SCPPBENU data set that is provided through the ServerPac RECEIVE process.

For z/OS orders, GENSKEL processing can take as much as 30 minutes or longer to complete. Subsystem orders might need only several minutes to complete.

The installation jobs should be submitted in sequence. Always read the DOC section before you select and submit the related jobs. All installation steps and jobs are also described in *ServerPac: Installing Your Order*.

File-tailored jobs might already exist in the SCPPBENU data set. Set the Replace Job field to Y to replace jobs; set the field to N to preserve them.

Pressing the Enter key displays the panel shown in Figure 4-36 on page 198.

## 4.36 Generate installation jobs

```
CPPP6127 ----- Pre-Process Installation Jobs -----
COMMAND ==>

GENERATE JOBSTREAM

  Enter JOBCARDS

> //GENSKEL  JOB (accounting information),          <
> //          'WAYNE O' 'BRIEN',NOTIFY=&SYSUID,      <
> //          CLASS=A,MSGCLASS=K,MSGLEVEL=(1,1),REGION=4M <
> /** ----- <

Installation  ISPLLIB ==> ISP.SISPLOAD
              ==>
              ISPLIB ==> ISP.SISPMENU
              ==>
Libraries     ISPPLIB ==> ISP.SISPPENU
              ==>
              ISPSLIB ==> ISP.SISPSENU
              ==>
              ISPTLIB ==> ISP.SISPTENU
              ==>
```

Figure 4-36 Pre-Process Installation Jobs

Before you submit the generate job, you must verify the JOB statement and the ISPF system libraries to be used, as follows:

ISPLLIB	Name of your SISPLLIB data set
ISPLIB	Name of your SISPLIB data set
ISPPLIB	Name of your SISPPLIB data set
ISPSLIB	Name of your SISPSLIB data set
ISPTLIB	Name of your SISPTLIB data set

You must provide at least one data set name for each ISPF library type. The default GENSKEL job assumes that you have SYSDA defined. If you do not, you must modify the JCL to specify a volume serial or an available esoteric or generic device name for your system.

When you submit a job for execution, the job number is written to a processing log. You can capture the job output by having it written to data set SCPPOENU. Doing so requires that you set two installation variables as follows:

- ▶ Set variable synonym OUTPUT LOGGING to YES.
- ▶ Set variable synonym JOBNAME to the TSO/E userid of the user who selects the jobs with the 0 command. (TSO OUTPUT copies the joblog from spool, which requires the jobname to be the USERID plus a character).

If you enter the SUMMARY primary command at the Job Selection List, the Processing Log panel in Figure 4-37 on page 199 is shown.

## 4.37 Displaying the processing log

```
CustomPac ----- Installation JOBS ( 0S261466 ) ----- ROW 1 TO 5 OF 5
COMMAND ==> SCROLL ==> HALF

Processing LOG

PRIM Cnds:(? SET L F N P SORT)
LINE Cnds:(Output)

      S  STEPname  JOB name job ID      RC  UserID  DATE stamp
      -  - - - - -  - - - - - - - - - -  - - -  - - - - -
      $IDCAMS  ST0B4DCT JOB04090  0012 WSOBRIEN  97/09/01 09:06:23
      OFFLINIT ST0B4A00 JOB04092  0000 WSOBRIEN  97/09/01 10:08:46
      DEFCAT   ST0B4D00 JOB04097  0000 WSOBRIEN  97/09/01 13:21:12
***** BOTTOM OF DATA *****
```

Figure 4-37 Processing Log Panel: SUMMARY Command

This panel lists the jobs that were submitted from the job stream and their respective return codes. If output logging is active, you can browse the job output. After a job's completion, the job output can be seen using the **Output** line command.

The job copying data sets to SystemPac Vols (RESTORE) may run for a long time, depending on the number of products your ServerPac order contains. You should have two tape drives and all the tape cartridges shipped with your order available before you start the RESTORE job.

Post-installation and customization is product- and installation-dependent, and should be related to your specific requirements.

After the installation jobs have completed, you should be able to IPL and test your new z/OS system.

## 4.38 Save configuration panel

```
CPPP6041 ----- SAVE Configuration ( 0S261466 ) -----
COMMAND ==> _

Specify SAVE Library

Enter the High Level Qualifier of the Library to which the
Order Configuration will be Saved

      ==> SYSPOK.OS261466.CONFIG

The default qualifier used is 'OrderHLQ'.
You may enter a Comment to identify the Configuration. This
is recommended if you use a qualifier other than the default.

      ==> DEBBIE'S Z/OS REL.1 SET-UP WITH RECOMMENDED LAYOUT

MASTER HLQ is : SERVRPAC.MASTER
```

Figure 4-38 Save Configuration panel

After you install a ServerPac order, you can use the Save Used Configuration function of the dialog to save your work configuration. Doing so can help you save time in installing subsequent ServerPac orders. Rather than manually re-entering all of the data required for each new order, you can merge the saved configuration with the new order and avoid much of the data entry.

Specify the HLQ for the configuration. The configuration data set is appended with either of the following low level qualifiers:

- ▶ SCPPSENU: For skeleton libraries
- ▶ SCPPTENU: For table libraries

If the libraries do not exist, the dialog prompts you to confirm the libraries can be allocated. If the libraries already exist, a new panel is shown to you to confirm the deletion of the old libraries.

As the last step of the installation, you should update the inventory by entering a **U** on the Installation Menu panel.

# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

## IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 202. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *z/OS Version 1 Release 3 and 4 Implementation*, SG24-6581
- ▶ *z/OS Version 1 Release 2 Implementation*, SG24-6235
- ▶ *z/OS V1R3 DFSMS Technical Guide*, SG24-6569
- ▶ *OS/390 Version 2 Release 10 Implementation*, SG24-5976

## Other publications

These publications are also relevant as further information sources:

- ▶ *IBM ServerPac Using the Installation Dialog*, SA22-7815
- ▶ *z/OS and z/OS.e Planning for Installation*, GA22-7504
- ▶ *z/OS Hardware Configuration Definition: User's Guide*, SC33-7988
- ▶ *z/OS Hardware Configuration Definition Planning*, GA22-7525
- ▶ *z/OS TSO/E Command Reference*, SA22-7782
- ▶ *z/OS TSO/E Customization*, SA22-7783
- ▶ *z/OS MVS Initialization and Tuning Reference*, SA22-7592
- ▶ *Interactive System Productivity Facility Getting Started*, SC34-4440

## Online resources

These Web sites and URLs are also relevant as further information sources:

- ▶ z/OS marketing and service withdrawal dates:  
[http://www.ibm.com/servers/eserver/zseries/zos/support/zos\\_eos\\_dates.html/](http://www.ibm.com/servers/eserver/zseries/zos/support/zos_eos_dates.html/)
- ▶ Samples of ServerPac Installing Your Order (IYO):  
<http://www.ibm.com/servers/eserver/zseries/zos/installation/#pubs>
- ▶ z/OS marketing and service withdrawal dates:  
[http://www.ibm.com/servers/eserver/zseries/zos/support/zos\\_eos\\_dates.html/](http://www.ibm.com/servers/eserver/zseries/zos/support/zos_eos_dates.html/)
- ▶ z/OS Hot Topics Newsletters:  
[http://www.ibm.com/servers/s390/os390/bkserv/hot\\_topics.html](http://www.ibm.com/servers/s390/os390/bkserv/hot_topics.html)
- ▶ List of vendors who support z/OS.e:  
<http://www.ibm.com/servers/eserver/zseries/solutions/s390da/r13e.html/>

- ▶ Multimedia presentation describing z/OS.e:  
<http://www.ibm.com/servers/eserver/zseries/zose/>
- ▶ z/OS.e documentation:  
<http://www.ibm.com/servers/eserver/zseries/zose/bkserv>
- ▶ General z/OS and z/OS.e information:  
<http://www.ibm.com/servers/eserver/zseries/zos/>
- ▶ General z/OS software prerequisites, coexistence, release migrations, and fallback information:  
[http://www.ibm.com/servers/eserver/zseries/zos/bkserv/find\\_books.html/](http://www.ibm.com/servers/eserver/zseries/zos/bkserv/find_books.html/)

## How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

[ibm.com/redbooks](http://ibm.com/redbooks)



**ABCS of z/OS System Programming Volume 1**

(0.2" spine)  
0.17" x 0.473"  
90 x 249 pages









# ABCs of z/OS System Programming Volume 1

**Introduction to z/OS  
and storage concepts**

**TSO/E, ISPF, JCL, and  
SDSF**

**z/OS delivery and  
installation**

The ABCs of z/OS System Programming is a ten 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 that you need to start your research into z/OS and related subjects. If you would like 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.

Volume 1 provides an understanding of the software and zSeries architecture and how it is used together with the z/OS operating system. This includes the main components of z/OS needed to customize and install the z/OS operating system.

Specific topics covered in this volume are:

- Introduction to the products and components that make up a z/OS system
- z/OS storage concepts
- The main components needed to install and customize z/OS: TSO/E, ISPF, JCL, and SDSF
- The z/OS delivery options and the download process using the ServerPac option

**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](http://ibm.com/redbooks)**

SG24-6981-00

ISBN 073849934X