

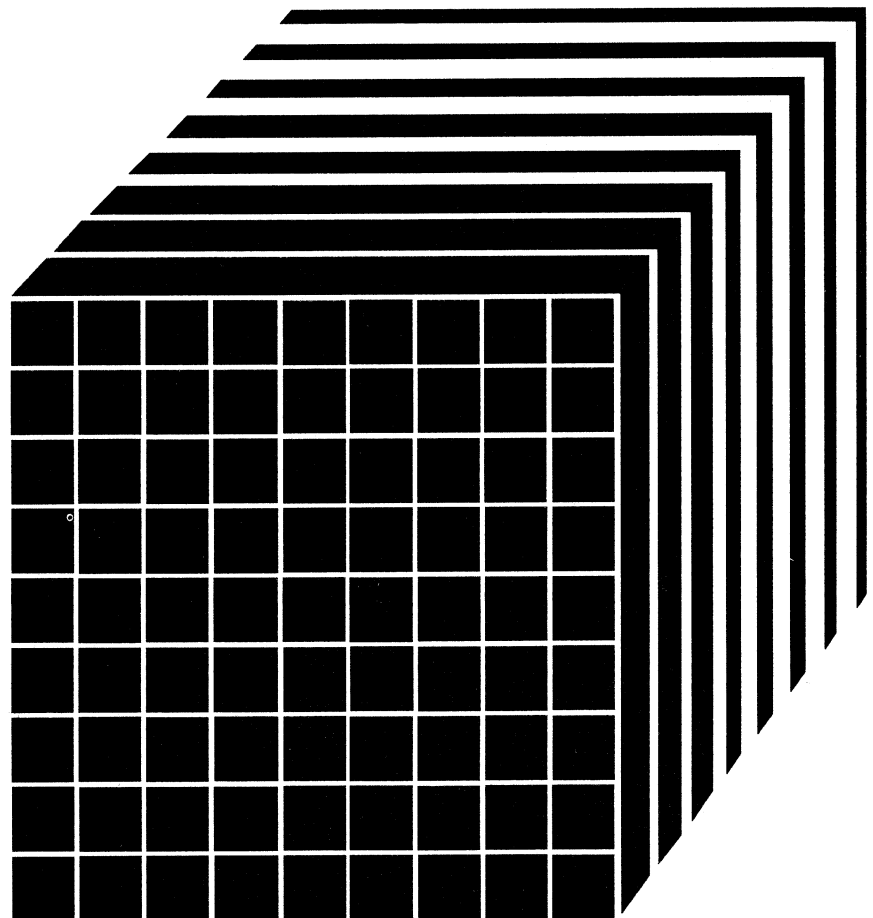


Virtual Machine/ System Product

Introduction

Release 5

GC19-6200-4



Fifth Edition (December 1986)

This edition applies to Release 5 of the IBM Virtual Machine/System Product (VM/SP), program number 5664-167, and to all subsequent releases and modifications until otherwise indicated in new editions or Technical Newsletters. Changes are made periodically to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest *IBM System/370, 30xx and 4300 Processors Bibliography*, GC20-0001, for the editions that are applicable and current.

This edition is a revision of GC19-6200-3. Because readers are not expected to be familiar with past editions of the book, revision bars are not used to mark new or changed information.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent program may be used instead.

Ordering Publications

Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality. Publications are *not* stocked at the address given below.

A form for reader's comments is provided at the back of this publication. It can also be used to comment on the VM/SP online HELP facility. If the form has been removed, comments may be addressed to IBM Corporation, Information Development, Department G60, P.O. Box 6, Endicott, NY, U.S.A. 13760. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Preface

This book describes, at an introductory level, what Virtual Machine/System Product (VM/SP) is and what it can do for users. It is meant for anyone who wants basic information about VM/SP and its capabilities. If you want to learn how to **use** VM/SP, you should begin with *VM/SP CMS Primer* as a guide.

You might use the information in the *VM/SP Introduction* if you:

- Plan a VM/SP installation
- Now know about VM/SP's capabilities, but would like to know more about VM/SP
- Need background knowledge for using licensed programs with VM/SP
- Migrate to VM/SP from another type of operating system
- Need basic knowledge about VM/SP for any reason.

None of the information in this book is meant to be used as a guide or reference for **using** VM/SP. Each topic is discussed only to a depth needed to introduce VM/SP and give references to more in-depth information.

How to Find More Information

For many people the most difficult thing about learning a computer system, especially a large one, is simply finding the necessary information. Each topic in this book includes references that tell you where to find more information about that topic. The last chapter of this book will help you find information in other books of the VM/SP library.

Many discussions in this book include references, either direct or indirect, to specific commands associated with that topic. One reason is that a command name makes a good reference word for locating additional information in another manual.

VM/SP General Information, GC20-1838, is a broader, less technical introduction to VM/SP. *VM/SP Library Guide, Glossary, and Master Index*, GC19-6207, is the complete guide to finding information in the VM/SP library.

What You Should Know Before Reading this Book

Before you use this book, you should know:

- The basics of data processing
- Operating systems fundamentals
- Data processing terminology.

This book assumes that you know the meanings of words such as “processor,” “system,” and “storage.” The book assumes, however, that concepts such as “paging,” “spooling,” and “networking” may be new to you.

Some sources for background information are:

- *VM/SP CMS Primer*, SC24-5236
- *Computing Systems Fundamentals Overview Text (self-study)*, GR29-0280
- *Introduction to IBM DP Systems (self-study)*, GC20-1684
- *Vocabulary for Data Processing, Telecommunications, and Office Systems*, GC20-1699.

About this Book's Content

The first page of each chapter has a brief overview of that chapter's contents, plus a table of contents for that chapter. An index is included at the back of the book.

You will probably find it best to read the whole book from start to end. Most topics assume you are familiar with ideas and terms discussed in earlier topics.

Because the information in this book reflects a system that is designed to help people at many levels do many kinds of work, some readers may elect to skip some of the topics. For instance, if you are a general user, not involved in system planning, generation, administration, or maintenance, you may find that the first two chapters of this book satisfy your needs. But if your job relates to the system itself, rather than with just its use, you should find the entire book useful.

How To Order Past Editions

To obtain editions of this book that pertain to earlier releases of VM/SP, you must order using the pseudonumber assigned to the respective edition. For:

Release 4, order GT00-1575

Release 3, order GT00-1349

Contents

Chapter 1. Some VM/SP Basics	1
A Short Introduction to VM/SP	3
The Virtual Concept	7
What a Virtual Machine Is	12
CP Creates Virtual Machines and Manages Real Resource Usage	16
Paging Allows Many Users to Share Processor Storage	20
Minidisks: Virtual Direct Access Storage	23
Spooling	26
VM/SP Is a Conversational System	33
A Summary of VM/SP Concepts	36
 Chapter 2. Putting VM/SP to Work	 37
Basics about Working in the Virtual Environment	39
Developing Application Programs under CMS	44
Debugging Application Programs	50
Handling Problems with VM/SP IPCS	54
Executing Programs under CMS	57
Processing Jobs in Batch Mode under CMS	61
Running Operating Systems in a Virtual Machine	65
Some Other Programs That Run under VM/SP	67
 Chapter 3. A Closer Look at CP	 71
Communicating with CP	73
Defining and Changing Virtual Machine Configuration and Status	77
Inter-User Communications Vehicle (IUCV)	82
Hardware Devices That Handle VM/SP Input/Output	85
Assuring Data Integrity	87
Performance Options Can Affect VM/SP Execution	90
Diagnosing System Programming Problems	95
 Chapter 4. A Closer Look at CMS	 97
Communicating with CMS	99
CMS from the User's Point of View	104
Command Language Processors under CMS	107
VM/SP System Product Editor	110
CMS File System	113
CMS Session Services	118
CMS HELP Facility	121
The Console Stack Helps Keep Work Flowing	124
IBM Enhanced Connectivity Facilities on VM/SP	127
 Chapter 5. Extending VM/SP's Capability	 129
Transparent Services Access Facility (TSAF)	131
Group Control System (GCS)	135
Network Products That Run on the Group Control System	137
RSCS Networking	141

VM/Interactive Productivity Facility	144
VM/Directory Maintenance	147
VM/Pass-Through Facility	150
Structured Query Language/Data System (SQL/DS)	154
VM/SP High Performance Option	157
 Chapter 6. Applications for VM/SP	 163
Application System (AS)	165
Professional Office System (PROFS)	168
Document Composition Facility (DCF)	171
DisplayWrite/370	175
Graphical Data Display Manager (GDDM)	180
The Information Facility (TIF)	183
Structured Query Language/Data System (SQL/DS) for Programmers ..	185
SQL/DS with Query Management Facility (QMF)	189
 Chapter 7. Programming Language Products	 193
OS/VS COBOL	195
PL/I	199
Assembler Language	203
VS FORTRAN	206
FORTRAN Interactive Debug	209
VS APL	211
IBM BASIC	214
Display Management System for CMS (DMS/CMS)	217
 Chapter 8. Distributed Data Processing Using VM/SP	 221
Why Distributed Data Processing?	223
The Pieces That Make Up a VM/SP DDP Network	226
Programmable Operator Reduces Network Operator Requirements ...	230
 Chapter 9. Connectivity, Personal Computers, and VM/SP	 235
IBM Enhanced Connectivity Facility	237
Why is there a need for Connectivity Facilities?	237
What Do You Need to Use the Connectivity Facility?	237
What You Can Do with Enhanced Connectivity Facilities	238
Reference	240
PC/VM Bond	241
Overview	241
Requirements	242
Reference	243
VM/PC (Virtual Machine/Personal Computer)	244
Overview	244
Requirements	245
Reference	245
 Chapter 10. VM/Integrated System	 247
 Chapter 11. Planning, Installing, and Servicing VM/SP	 253
Planning VM/SP	255
Installing VM/SP	258
Servicing VM/SP	260

Chapter 12. Where to Find More Information about VM/SP	265
Bibliography	273
Index	277

Figures

1.	CP and CMS Allow VM/SP Users to Share One System	4
2.	Virtual Storage May Be Much Larger Than Real Storage	8
3.	A Virtual Machine	12
4.	CP Creates a User's Virtual Machine	18
5.	CP Allocates Storage According to User Needs	21
6.	CP Protects Minidisks from Unauthorized Use	24
7.	Shared Use of the System Printer through Spooling	27
8.	Spooling Virtual Punch Output to Another User	29
9.	CP Commands May Be Issued to CMS	33
10.	Communicating with CP	41
11.	Virtual Machine Input and Output	43
12.	Compiling a High-Level Language Program	47
13.	IPCS Is an Organized Approach to Handling Software Problems ..	55
14.	Generating and Executing an Application Program	58
15.	Running Application Programs in the Batch Virtual Machine ...	62
16.	Loading Operating Systems into Virtual Machines	66
17.	Some Programs That Run in Virtual Machines	69
18.	A Command Class for Each Type of User	75
19.	Changing Virtual Machines	80
20.	Communication Paths For IUCV Functions	83
21.	CP Protects against Unauthorized System Use	87
22.	Performance Options Can Improve Productivity	91
23.	User/CMS Communication	101
24.	EXEC Procedures Simplify User-to-System Communication	108
25.	Fixed- and Variable-Length Records	114
26.	Using the Program Stack	125
27.	How RSCS Networking Transmits a File to a Remote Non-SNA node	142
28.	VM/IPF General Task Menu Panel	145
29.	Updating the VM/SP System Directory	148
30.	Using VM/Pass-Through to Access Another Network System ...	150
31.	GDDM May Be Used for Displaying a Variety of Graphics	182
32.	Examples of SQL Commands Embedded in Programs	187
33.	SQL/DS Responds to Either SQL or QBE Queries	191
34.	Programs for Servicing VM/SP	261

Chapter 1. Some VM/SP Basics

This chapter discusses VM/SP features that:

- Help users do a variety of data processing tasks
- Share system resources among many users
- Communicate with the people who use VM/SP.

Many of the concepts in this chapter will not be new to you. You will simply see them from a new point of view. For example, the words **virtual** and **spool** are familiar to most people. But, from a VM/SP point of view, the definitions of these words may differ from the more familiar meanings.

Chapter Contents

A Short Introduction to VM/SP	3
CP Is the Resource Manager	3
CMS Manages Work Flow	5
GCS Manages Subsystems	6
Licensed Programs Extend VM/SP's Capabilities	6
The Virtual Concept	7
Why We Need Virtual Storage	7
How Virtual Storage Differs from Real Storage	9
Virtual Unit Record Devices	9
Magnetic Tape Devices	10
Virtual Direct Access Storage Devices	10
How a Virtual Console Differs from a Real Console	10
How VM/SP Shares Processor Usage	11
What a Virtual Machine Is	12
Requirements for Using a Virtual Machine	13
Operating Systems Run in Virtual Machines	13
A Virtual Machine Includes Virtual I/O Devices	14
Some Other Facilities That a Virtual Machine Provides	14
CP Creates Virtual Machines and Manages Real Resource Usage	16
VM/SP System Directory Entries Define Virtual Machines	16
Changing the Virtual Machine	17
How CP Divides Real Processor Time among Virtual Machines ..	17
Real Storage Allocation Changes with User Needs	17
How CP Determines Virtual Storage Size	19
Virtual Machine Requests for Real Device Usage	19
Paging Allows Many Users to Share Processor Storage	20
How Virtual Storage and Real Storage Are Organized	20
Paging Is the Exchange of Virtual Storage between Real Storage and Direct Access Storage	21
Minidisks: Virtual Direct Access Storage	23
How CP Defines Minidisks	23
Shared Access to Minidisks	25
Minidisk Initialization	25
Spooling	26
How CP Manages Spooling	26
Spooled Virtual Punch and Virtual Reader Data	28
Spooling from the User's Point of View	29
Spooling from the Operator's Point of View	30
How to Retrieve Virtual Console Data through Spooling	31
A Summary of CP's Spooling Activity	32
VM/SP Is a Conversational System	33
How a General User Communicates to VM/SP	34
System Messages Communicate to Users	34
How the System Operator Communicates to VM/SP	35
A Summary of VM/SP Concepts	36

A Short Introduction to VM/SP

Overview

- VM/SP is an interactive, multiple-access operating system.
- Two major VM/SP components, CP and CMS, provide a variety of services.
- VM/SP can help a user:
 - Write, test, and debug programs
 - Process jobs in batch mode
 - Create and edit information files
 - Manage a share of system resources
 - Execute a variety of operating systems and application programs.
- Group Control System (GCS), another facility of VM/SP, manages subsystems that support a Systems Network Architecture (SNA) network.
- Many licensed programs are available for extending the capabilities of a VM/SP system.

VM/SP is an interactive, multiple-access operating system. Interactive means that there is two-way communication between users and VM/SP. Multiple-access means that several people can use a VM/SP system at the same time.

VM/SP has two basic components: Control Program (CP) and Conversational Monitor System (CMS). Each provides its own types of services. You should understand the descriptions of CP and CMS that follow before you read the rest of this book.

CP Is the Resource Manager

Control Program (CP) manages system resources and gives each person using the system an individual working environment. The resources that CP manages include:

- Processor functions
- Processor storage
- Input and output devices.

Some VM/SP Basics

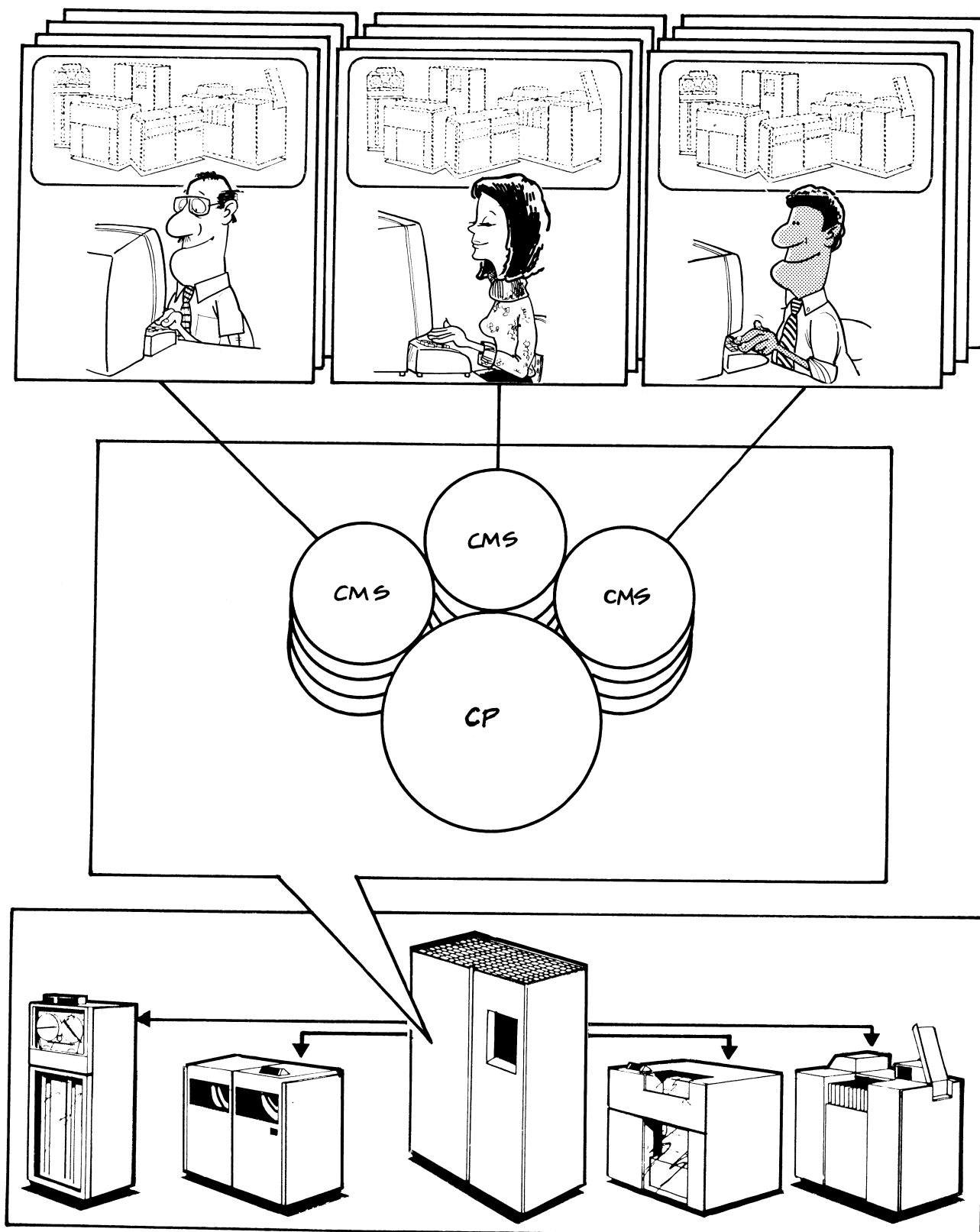


Figure 1. CP and CMS Allow VM/SP Users to Share One System

CP is the one part of VM/SP that is always required for system use. This is because CP creates the system work environment. CP controls the system resources that are available to the user during a work session (logged onto) on VM/SP.

To some extent, CP lets the user manage the portion of system resources it assigns to that user. For example, a user can get a resource status report from CP that shows such things as the direct access storage assigned to that user. CP can also make some changes to the resources it has allocated to a user.

CP lets many people share the use of system resources. It allocates resource usage in such a way that each person works independently of others on the system. In other words, CP manages one physical system so that it seems to give each of the users a separate and independent system. In Figure 1 on page 4 each person has the impression of being the only system user. Users often do not have visual contact with other users and may never see the real system they use.

When a user first logs onto VM/SP, only CP controls the working environment. Many of VM/SP's facilities are immediately available to the user. For example, the system operator does many system management tasks under control of CP. However, most of the work done on VM/SP requires the use of an operating system in the working environment to help with data processing tasks and to manage work flow. There are many operating systems that run under VM/SP.

CMS Manages Work Flow

Conversational Monitor System (CMS), although a component of the VM/SP operating system, is itself an operating system. Every VM/SP system includes CMS. CMS runs only with CP. As the name "conversational" implies, there is two-way communication between system users and CMS. Users talk to CMS with commands; CMS uses system messages to talk to users.

CMS can help the user do a wide variety of tasks. For example, the user can:

- Write, test, and debug application programs
- Create and edit information files
- Execute application programs
- Process jobs in batch mode
- Manage the VM/SP working environment
- Communicate with other users of the VM/SP system.

Some VM/SP Basics

As shown in Figure 1 on page 4, each user works with a separate copy of CMS.¹

Chapter 2 of this book discusses these and other uses for the CMS operating system.

GCS Manages Subsystems

The Group Control System (GCS) is a virtual machine supervisor that manages multiple tasks. Like CMS, GCS is shipped as a part of every VM/SP system. However, an installation may or may not choose to install GCS. Its main purpose is to manage subsystems that support a Systems Network Architecture (SNA) network.

A section in Chapter 5 of this book discusses GCS in more detail.

Licensed Programs Extend VM/SP's Capabilities

VM/SP facilities, together with separate licensed programs that run under VM/SP, provide many additional services. These services include such things as:

- Accessing other systems to do data processing
- Communicating with users on other systems
- Managing routine office tasks
- Diagnosing system program problems
- Making VM/SP easier to administer.

Reference

Chapters later in this book discuss special-purpose facilities and licensed programs, as well as facilities provided by CP and CMS.

¹ Each copy does not have to be a complete copy because of "segment sharing" supported by CP. This is discussed in the chapter describing CP.

The Virtual Concept

Overview

- Although many people can share the use of a VM/SP system, each person seems to have exclusive use of that system's resources.
- This concept applies to input and output devices as well as to processor functions and storage.
- VM/SP lets direct access storage simulate actual system devices and storage.
- The VM/SP control program (CP) manages real system resources and gives each user separate virtual system resources.
- Total virtual storage size can be much larger than the real storage size.

This topic explains how VM/SP (Virtual Machine/System Product) puts the virtual concept to work.

Although many people can use a VM/SP system at the same time, each of them may have the impression of being the only system user. VM/SP acts as a super traffic director, making sure each user gets a fair share of system resources. VM/SP shared resources include both processor and input or output devices. The system simulates actual resources in such a way that it is like having a separate and complete system available to each of its users. Each one has a simulated (virtual) machine to use. Thus the name "Virtual Machine."

Why We Need Virtual Storage

Before the idea of virtual storage was put to use, the upper limit of a computer program's address space was the size of processor storage. If a program was too large to run in a processor, it had to be written and run in pieces. This was a major problem for programmers.

In today's virtual systems it is possible that only a part of a program that is running can, at any given instant, be in the processor real storage. The total address space available to the program includes direct access device (virtual) storage as well as processor storage. Thus, for the virtual environment, storage address space is extended far beyond the limiting factor of processor storage size. Now, programmers may think more about a program's function and less about storage constraints during program execution.

Some VM/SP Basics

A VM/SP system can have many people logged on at the same time. Each of them may need storage space for an operating system plus a large work area. This means that each of these users could need one million or more storage positions (bytes). That adds up to a total of 100 million or more bytes of storage to support 100 users. This is more than the size of very large processor storage. Therefore, some of this storage must reside some place other than in the processor.

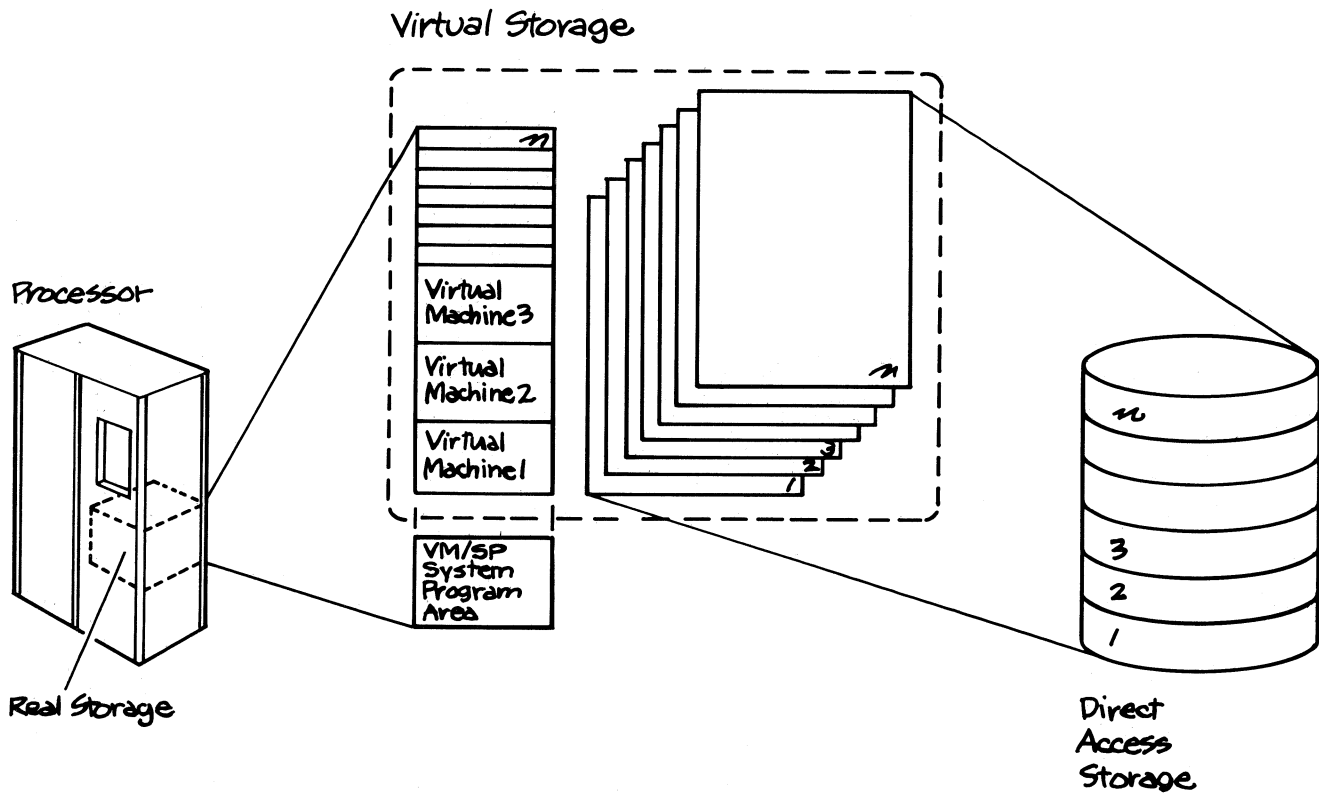


Figure 2. Virtual Storage May Be Much Larger Than Real Storage

If a program needs more storage than is available to it in the processor, CP uses direct access storage devices (DASD) for part of that storage. The most common kind of DASD is the disk. CP then moves portions of storage content to the processor when it is needed and returns it to direct access devices during the time it is not needed for processing. Because of this facility for having data reside in direct access storage until it is needed in the processor, the size of a user's virtual storage can be very large, even larger than the entire processor storage.

How Virtual Storage Differs from Real Storage

Virtual storage can be on a direct access storage device when the contents are not required for processing. When virtual storage content is required for processing, it is moved to real storage. A given area of real storage may serve as virtual storage for any one of the system users, depending on the relative processing needs of the users at that time. This allows a relatively small amount of real storage to serve as processing storage for what is collectively a much larger virtual storage, as shown in Figure 2 on page 8.

A user's virtual storage size can be any multiple of 4K, with a minimum of 8K. (4K stands for four thousand. This is rounded from 4096 bytes.) The maximum is limited only by the addressing structure of the processor on which VM/SP runs and by availability of direct access storage. On processors that use System/370 architecture, addresses are three bytes long. The largest hexadecimal value that can be expressed in three bytes is about 16 million (16 meg). Therefore, in these systems the upper limit of virtual storage is 16 megabytes.

Real storage, also called main storage, is a part of the system processor. Real storage size is determined by the processor model installed. Each real storage position has a constant address, ranging from 0 to the maximum processor storage size. CP adjusts (maps) virtual addresses to real addresses as needed.

Virtual Unit Record Devices

This book uses the term "unit record devices" to refer to readers, punches, and printers.² These devices, regardless of their speed, can accommodate only one user at any given instant. However, it is possible that several users will simultaneously need the services of the same type of input or output device. It is not practical for a system to have enough of these devices to respond immediately to all input and output requests. On the other hand, users cannot be expected to stop working until an input or output device becomes free.

VM/SP's answer to this problem is to simulate device functions by spooling (storing) the data on direct access storage until the required real device becomes free to process the data. It holds the data on direct access storage until the receiving user or device is ready to retrieve it. Meanwhile, the user is free to go about another task. This method of storing input or output data on direct access storage is how CP provides separate virtual system devices for each user.

² Historically, unit record devices were real card readers, card punches, and printers. But with changes in technology systems over the years, these have become 'virtual' card readers, punches and printers (the real ones are usually in the computer room connected directly to the computer system). Today, they are still referred to as virtual card readers, punches, and printers and collectively as 'unit record devices'.

Some VM/SP Basics

Magnetic Tape Devices

Because of the way tape is scanned, read, and written, having more than one person use the same tape device at the same time is not possible. If a magnetic tape device is needed, the user requests that it be assigned (dedicated) temporarily for that user's exclusive use. The system operator controls the dedicated usage of tape devices.

Virtual Direct Access Storage Devices

One direct access storage device (DASD) can meet the needs of many users. CP divides DASD storage into areas called "disks" or "minidisks." These minidisks vary in size to accommodate user or system needs. At any given time many users can be accessing the various minidisks that reside on one direct access storage device. However, each user usually is unaware of sharing the device with other users.

Normally, every VM/SP user has exclusive use of at least one minidisk to hold personal data files. These files are normally accessible only by that user. However, each user may share other minidisks, usually on a read-only basis, with other users.

CP authorizes access to minidisks when a user logs onto VM/SP. An entry in the system directory³ tells CP the size and location of each user's minidisks. There is more information about this subject later in this chapter under "Minidisks: Virtual Direct Access Storage."

How a Virtual Console Differs from a Real Console

The word "console" as used in this book means a device through which a user manages system functions. This differs from "terminal" in that a terminal can be used for any exchange between a user and the system. A console and a terminal may be the same physical device. A terminal becomes a console when it is used to manage virtual or real resources. Broadly speaking, there are two kinds of console users.

- System operator - manages real system resources
- General user - manages virtual machine operation.

The main difference between the system operator's virtual console and the general user's virtual console is in the authority CP gives each to issue certain commands.

The real system console is a part of the processor hardware. Only the system operator and certain maintenance people normally use the real console. They use it to take some action that affects the real system. For

³ The system directory is a table of information located on CP's direct access storage. It is built as part of the VM/SP system generation process. There are entries in the system directory that tell CP how to allocate system resources to each user.

example, the operator uses it to start the system, or a service representative might use it to diagnose processor problems.

Most users, including the system operator, manage their view of the system through a virtual console. A virtual console is a terminal through which a user logs onto VM/SP. Entries in the system directory for each user logging onto VM/SP determine the limits of that user's console functions.

How VM/SP Shares Processor Usage

A VM/SP system can share processing among several users through a process called "time slicing." The system does not run several programs at the same time. What it does is to run part of one program for a short time, then run part of another, and then another, until it returns to the first program and begins the cycle again. It appears to the person running each of these programs that no one else is using the processor.

Reference

Introduction to Virtual Storage in System/370, SR20-4260, describes the basics of virtual storage.

VM/SP Terminal Reference, GC19-6206, lists the terminal types that can be used to access VM/SP and describes virtual console options and procedures.

Some VM/SP Basics

What a Virtual Machine Is

Overview

- A virtual machine consists of a real terminal (virtual console), processor, direct access virtual storage, virtual input and output devices, and an operating system.
- A virtual machine provides functions that are very much like those of a real single-user system.
- CP creates a virtual machine (system environment) when a user logs onto VM/SP.
- Each virtual machine user has access to a share of each real system resource.

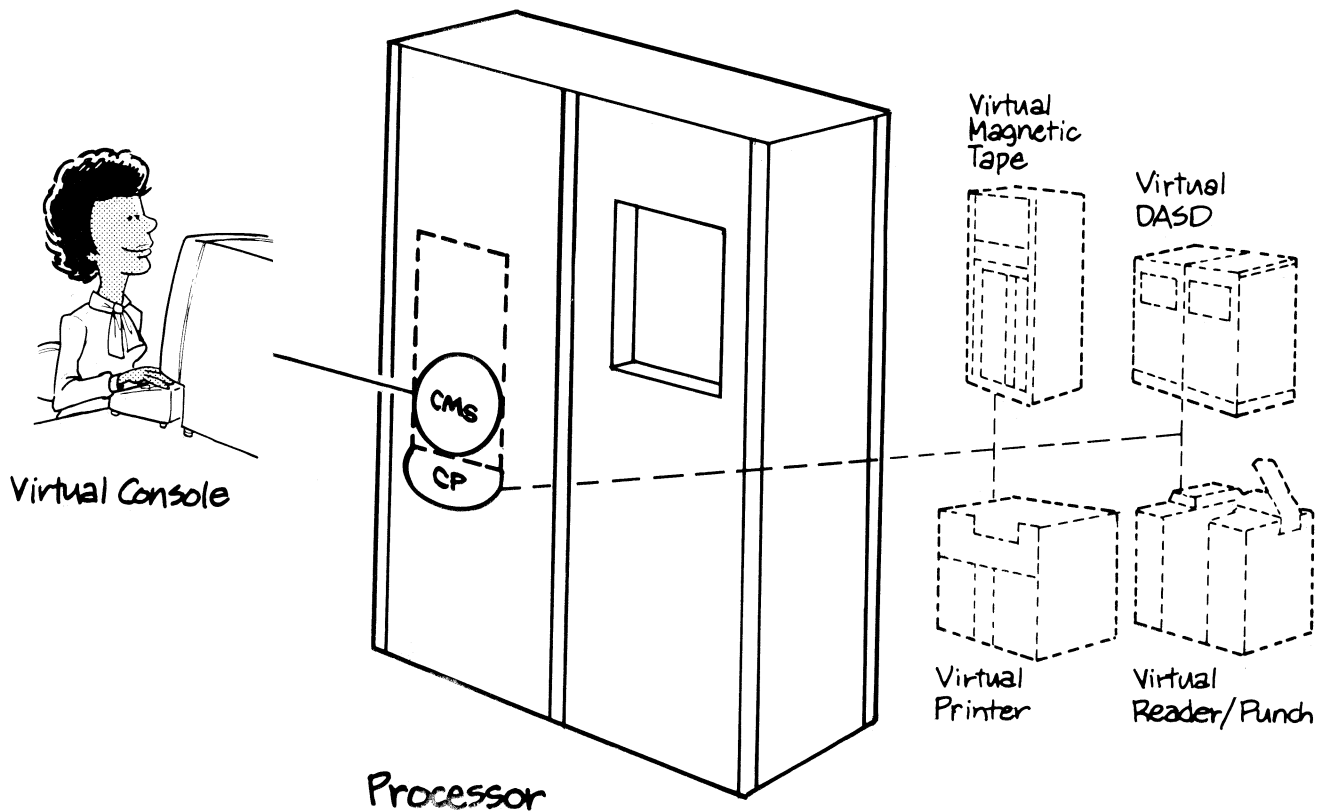


Figure 3. A Virtual Machine.

The facilities that make up a virtual machine are shown in Figure 3. The only one of these facilities that is normally visible to the user is the terminal (virtual console).

Requirements for Using a Virtual Machine

CP creates a virtual machine for a user's exclusive use if that user:

- Uses a supported terminal device connected to the system
- Is entered as a user in the system directory
- Completes the system logon procedure.

After CP creates a virtual machine, the user can work on it as though the user were the only person using the real system. The user simply loads any one of the many operating systems that run in a virtual machine, then goes to work. A virtual machine gives the user access to:

- A share of processor time
- Virtual storage
- Virtual unit record devices (reader, punch, and printer)
- Virtual direct access storage; shared minidisks or minidisks dedicated to the user's virtual machine
- Magnetic tape devices
- Other less frequently used resources. For example, communication lines to remote sites or adapters to connect processor channels to each other.

Operating Systems Run in Virtual Machines

CMS is the operating system most frequently run in a virtual machine. Other operating systems that run in a virtual machine include:

- VSE/SP (and earlier versions of this DOS based family of products)
- MVS/SP (and earlier versions of this OS based family of products)
- Remote Spooling Communications Subsystem Networking (RSCS)
- VM/SP itself.

To find more information about running operating systems under VM/SP, see "Reference" on page 15.

Some VM/SP Basics

A Virtual Machine Includes Virtual I/O Devices

Unit Record Devices

A virtual machine's unit record (reader, punch, and printer) devices are simulated by CP through use of direct access storage. From each user's point of view it is as though that user has exclusive use of the real system devices. In reality the user may be sharing the devices with one hundred or more other users. There is more information about this process later in this chapter under "Spool Space Is Temporary Direct Access Storage for Virtual Machine Input or Output Data."

Direct Access Storage Devices

CP shares direct access device storage among all virtual machines by dividing the DASD surface into areas called "minidisks." Minidisks are also called virtual direct access storage. Minidisks are discussed in a separate topic later in this book. Do not confuse the concept of sharing DASD among virtual machines with the term "shared DASD," which for some operating systems may mean sharing direct access storage devices among multiple processors.

Magnetic Tape Devices

Magnetic tape devices are not normally a permanent part of a virtual machine's configuration. A system operator can use the ATTACH command to request that CP reserve (dedicate) a tape device for exclusive use by a specific virtual machine. This is normally done for tape or direct access storage devices, but other devices are sometimes dedicated as well. Only the operating system running in a virtual machine can use a device dedicated to that virtual machine.

Some Other Facilities That a Virtual Machine Provides

CP can make several facilities, other than those already discussed in this topic, available to a virtual machine. In addition to a console, basic unit record devices, virtual storage, and a share of the processor's time, CP can allocate to a virtual machine such facilities as a timer function or an accounting function. CP can also allocate more than one of each of the unit record devices, or dedicate device types not commonly included in a virtual machine.

System directory entries or user commands to CP authorize the use of these facilities.

Communicating With Other Virtual Machines

Users can exchange messages or data files with other users in virtual machines on the same system. In addition, one virtual machine can share

data files residing in another virtual machine. Normally, the user is limited to reading data from those files. There is more about accessing minidisks later in this chapter under "Minidisks: Virtual Direct Access Storage."

User programs can communicate with user programs on the same system or on different systems. Refer to "Advanced Program-to-Program Communication/VM (APPC/VM)" and "Transparent Services Access Facility (TSAF)" in this manual for more information.

Requesting Status Information

Virtual machine users can display status information about their virtual machines or the VM/SP system. The QUERY command requests CP to display at a virtual console a wide range of information about the status of virtual machine resources. For example, it can indicate if a virtual reader is currently holding any files.

The INDICATE command requests from CP a status report about current usage of major resources provided by that VM/SP system. This may include status information about all users, or information that applies only to the user's virtual machine.

Reference

VM Running Guest Operating Systems, GC19-6212, gives system programmers detailed information about generating and using operating systems under VM/SP.

VM/SP CP Command Reference, SC19-6211, describes CP commands, related to virtual machine management, available to the system operator.

VM/SP Planning Guide and Reference, SC19-6201, describes system directory entries for allocating resources and facilities to virtual machines.

CP Creates Virtual Machines and Manages Real Resource Usage

Overview

- VM/SP system directory entries define each user's virtual machine.
- When a user logs onto VM/SP, CP provides that user with a virtual machine as defined by the system directory entries.
- CP commands allow a virtual machine to be further defined.
- CP allows each virtual machine use of processor resources at calculated intervals.
- CP makes the best use of processor storage by:
 - Allocating it to users on a demand basis
 - Moving the contents of inactive portions of processor storage to direct access storage.
- CP processes virtual machine requests to use real input or output devices.

VM/SP System Directory Entries Define Virtual Machines

When a user logs onto VM/SP, CP checks the system directory for a valid user identification (userid) for that user. If CP does not find the required userid in the directory, it rejects the attempt to log onto VM/SP.

For successful logon attempts, CP uses system directory entries to define virtual machines. System directory information includes:

- Virtual input/output device data
- Classes of VM/SP commands a user is authorized to issue
- Normal and maximum virtual storage sizes for each user
- Passwords that allow users to access the system
- Data to help account for use of system resources
- User priority for using system resources
- National language to be used in communications with the user.

Changing the Virtual Machine

CP can change some of the effects of the system directory entries on a virtual machine definition, based on commands issued by the user. This might include how output should be printed, or establish a link between a virtual machine and a minidisk in another virtual machine. These commands to CP can be issued either manually or automatically from procedures. In Figure 4 on page 18, CP receives directory statements and user commands as input for creating the user's virtual machine.

How CP Divides Real Processor Time among Virtual Machines

VM/SP allocates processor time to virtual machines on a time slicing basis. "Time slicing" means that each virtual machine receives a calculated share of the real processor's time. This is not to say that all virtual machines receive equal amounts of processor time. For example, virtual machines that operate in a conversational mode receive shorter but more frequent slices of time than do virtual machines in which application programs run.

At the end of a virtual machine's processor time slice, CP determines the type of processing that has been taking place. To do this, CP gathers data about virtual machine console request and interrupt activity. Based on this data, CP calculates when and for how long to return processor usage to each virtual machine. CP gives a virtual machine control of the processor only if that virtual machine is ready to process. CP does not give control of the processor to a virtual machine that is waiting for certain events to occur, such as execution of input or output operations.

A virtual machine's priority for gaining control of the processor can be influenced by including a priority value in that virtual machine's system directory entry. This value, controlled by the system administrator, is set at system generation time, but it can be changed later by the VM/SP system operator.

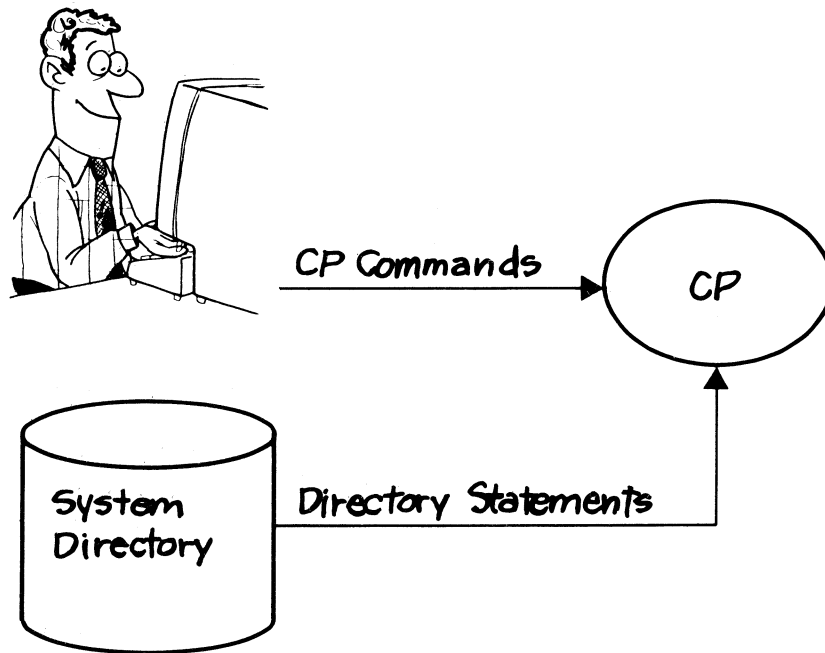
Real Storage Allocation Changes with User Needs

Depending on a virtual machine's needs, CP allocates more or less real storage to that virtual machine. During times of heavy processor activity, such as running a sort program, CP gives a virtual machine relatively more use of real storage, as illustrated in Figure 5 on page 21. On the other hand, jobs like text editing can decrease the real storage allocated to a virtual machine. The distribution of real storage depends on the relative needs among the virtual machines at any given time.

CP's management of storage transfer between real storage and direct access storage is discussed later in this chapter. See "Paging Allows Many Users to Share Processor Storage" on page 20.

Some VM/SP Basics

INPUT



RESULT

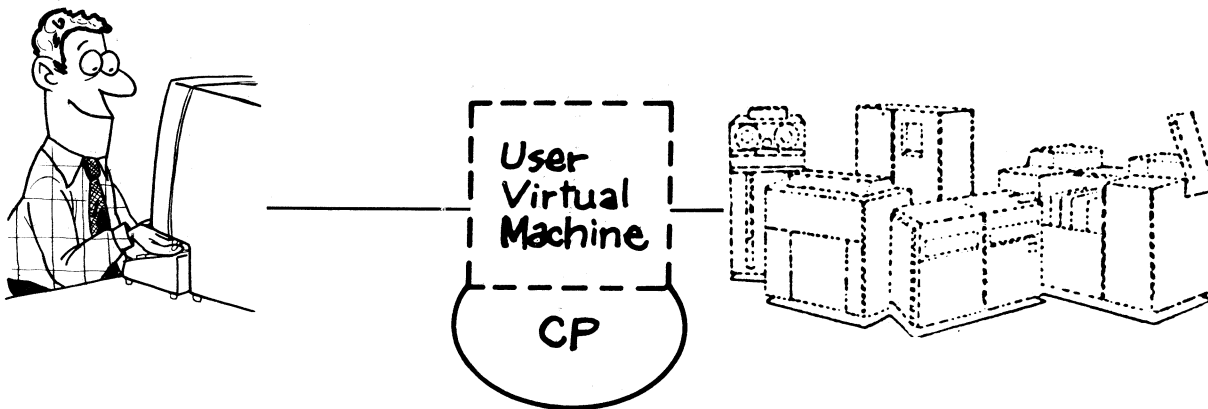


Figure 4. CP Creates a User's Virtual Machine

How CP Determines Virtual Storage Size

A VM/SP directory entry contains the normal and maximum virtual storage sizes for each virtual machine. When a user logs onto VM/SP, CP assigns the “normal” virtual storage size to the virtual machine it creates for that user. CP will honor the user’s request later in the session to change the virtual storage size (up to the maximum value specified in the system directory entry for that virtual machine).

Virtual Machine Requests for Real Device Usage

Each virtual machine can request output services as though it had exclusive use of system resources. A real system device can process only one of these requests at a time. Therefore, CP uses a buffering facility called “spooling” to hold output data waiting to be processed on a real device. CP manages the actual processing by real system devices of spooled files.

Reference

VM/SP Planning Guide and Reference, SC19-6201, describes definition of virtual machines in the VM/SP system directory.

VM/SP CP for System Programming, SC24-5285, discusses changing the VM/SP system directory and gives more in-depth information about virtual storage management.

Paging Allows Many Users to Share Processor Storage

Overview

- The basic unit of virtual storage content is 4096 bytes called a page.
- The number of virtual storage pages is usually much greater than the number of real storage page frames available. CP keeps this excess of virtual storage pages on direct access storage.
- CP moves pages from direct access storage to the processor as needed (“demand paging”) and assigns real storage page frames to virtual machines as needed.

How Virtual Storage and Real Storage Are Organized

Virtual storage is made up of 64K-byte areas called segments. CP subdivides segments of virtual storage into 4K-byte blocks called pages. When CP moves virtual storage from a direct access device into real storage, each page fits into a 4K-byte area of real storage called a page frame. In Figure 5 on page 21, CP transfers page frames from a relatively inactive virtual machine to a relatively active one. To help manage page and segment activity in a system, CP maintains a separate set of page and segment data tables for each active (logged-on) virtual machine.

Virtual storage addresses consist of a segment number, a page number, and a displacement within that segment. During program execution, CP changes the segment number, page number, and displacement of virtual storage addresses to real (absolute) storage addresses. This process is called “dynamic address translation.” Thus, CP can place pages of any user’s virtual storage into any available real storage frames.

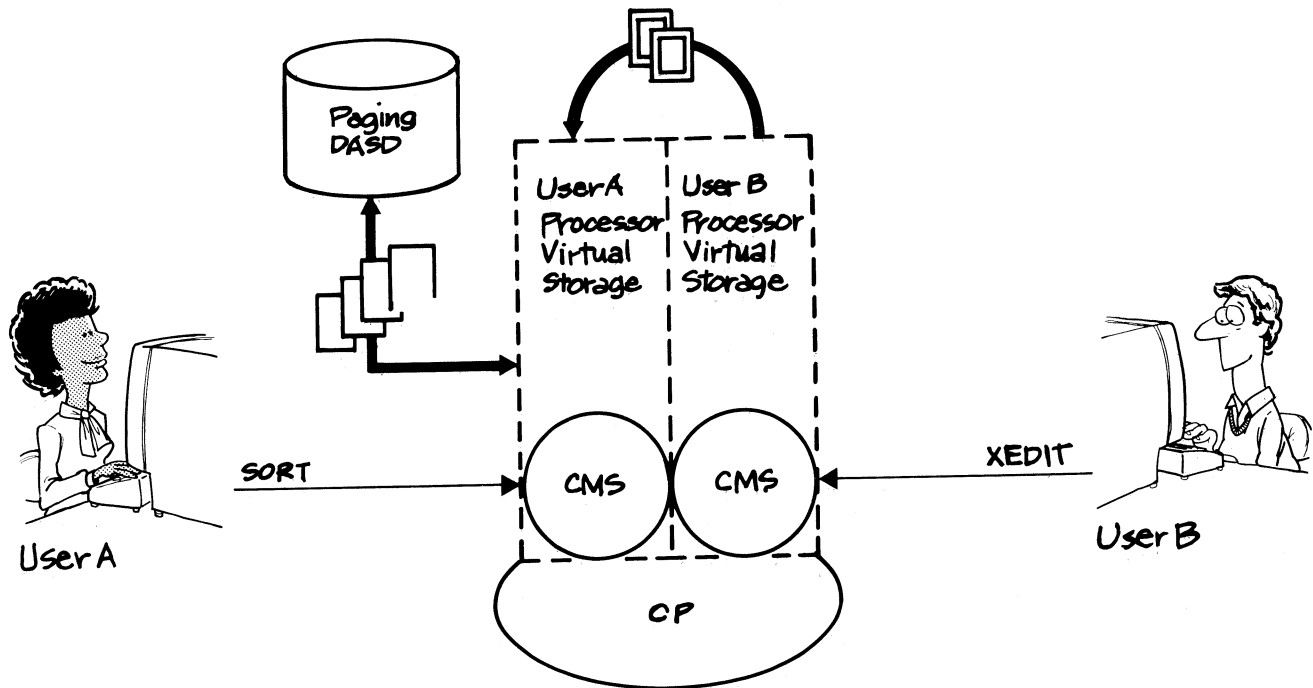


Figure 5. CP Allocates Storage According to User Needs.

Paging Is the Exchange of Virtual Storage between Real Storage and Direct Access Storage

To make better use of real storage, CP moves inactive (or relatively inactive) pages onto direct access storage. CP moves these pages back to real storage when they are needed for program execution. This movement of pages between real storage and direct access devices is called "paging."

CP manages paging operations with no action required of VM/SP users. The paging process is so fast that users are usually not aware that it is happening. CP reduces the effect on system performance by selectively paging out the most infrequently used storage. Pages that have been inactive are less likely to be needed by a future program event. Pages that a program refers to more often remain in real storage.

Once moved to direct access storage, a page remains there until it is needed by a program. The term "page fault" refers to the case where a page on direct access storage is needed by a program. A page fault results in CP's returning that page to real storage (paging in). CP does not try to anticipate when a particular page might be needed in real storage. This process of moving storage pages to the processor only as needed is called "demand paging."

Because of CP's ability to page storage to and from a direct access storage device, a processor can satisfy the needs of many active system users whose total virtual storage requirements far exceed available real storage.

Some VM/SP Basics

Areas on a direct access device to which paged data can be written or read with the least physical motion within the device are called “preferred” paging areas. Normally, pages that programs refer to more often reside in the preferred paging areas. If, however, either the preferred or nonpreferred paging area begins to fill, CP shifts pages to create a more balanced page storage. If necessary, the system operator can use the MIGRATE command to ask CP to move pages between preferred and nonpreferred areas. This relocation of pages is called “page migration.”

Reference

VM/SP CP for System Programming, SC24-5285, discusses in more detail paging and page migration from a system programmer’s point of view.

VM/SP Operator’s Guide, SC19-6202, describes commands that are available to the system operator for managing page storage.

Introduction to Virtual Storage in System/370, SR20-4260, describes in detail paging concepts and the relationship of virtual storage to real storage.

Minidisks: Virtual Direct Access Storage

Overview

- A minidisk (virtual disk) is a subdivision of consecutive cylinders on a real direct access storage volume.
- A minidisk can be defined for read/write access or read-only access.
- A minidisk can be defined for use by a virtual machine on a temporary or permanent basis.
- Each minidisk has a virtual disk address.
- A CMS minidisk is accessed according to a letter (filemode), A through Z. This letter establishes the order that CMS searches through minidisks when accessing files.

Each virtual machine under VM/SP uses direct access storage. Because the number of virtual machines normally exceeds the number of real direct access storage devices available, each virtual machine has exclusive use of only a portion (minidisk) of a real storage disk.

How CP Defines Minidisks

MDISK system directory statements initially define one or more permanent minidisks for each virtual machine. Operands on the MDISK statement specify such things as:

- DASD device type, virtual address, and volume serial number
- Minidisk size in cylinders or blocks
- Type of primary access mode (for example, read only) assigned to the minidisk
- Passwords that allow other virtual machine users to share this minidisk.

A direct access storage device has one real device address, but each minidisk on it has its own virtual address. The virtual address can be any value from 001 through 5FF (FFF if the ECMODE feature is on).

Some VM/SP Basics

Minidisks consist of consecutive cylinders. Each minidisk starts at virtual cylinder 0. Minidisks range in size from one cylinder (one block on FBA⁴ devices) to the entire disk volume. The storage capacity of a cylinder varies with the type of direct access storage device.

During a VM/SP session, users can request CP to allocate temporary minidisk space to their virtual machines. For example:

```
CP DEFINE T3380 AS 291 CYL 10
```

requests a 10-cylinder temporary minidisk at virtual device address 291 on an IBM 3380 device.

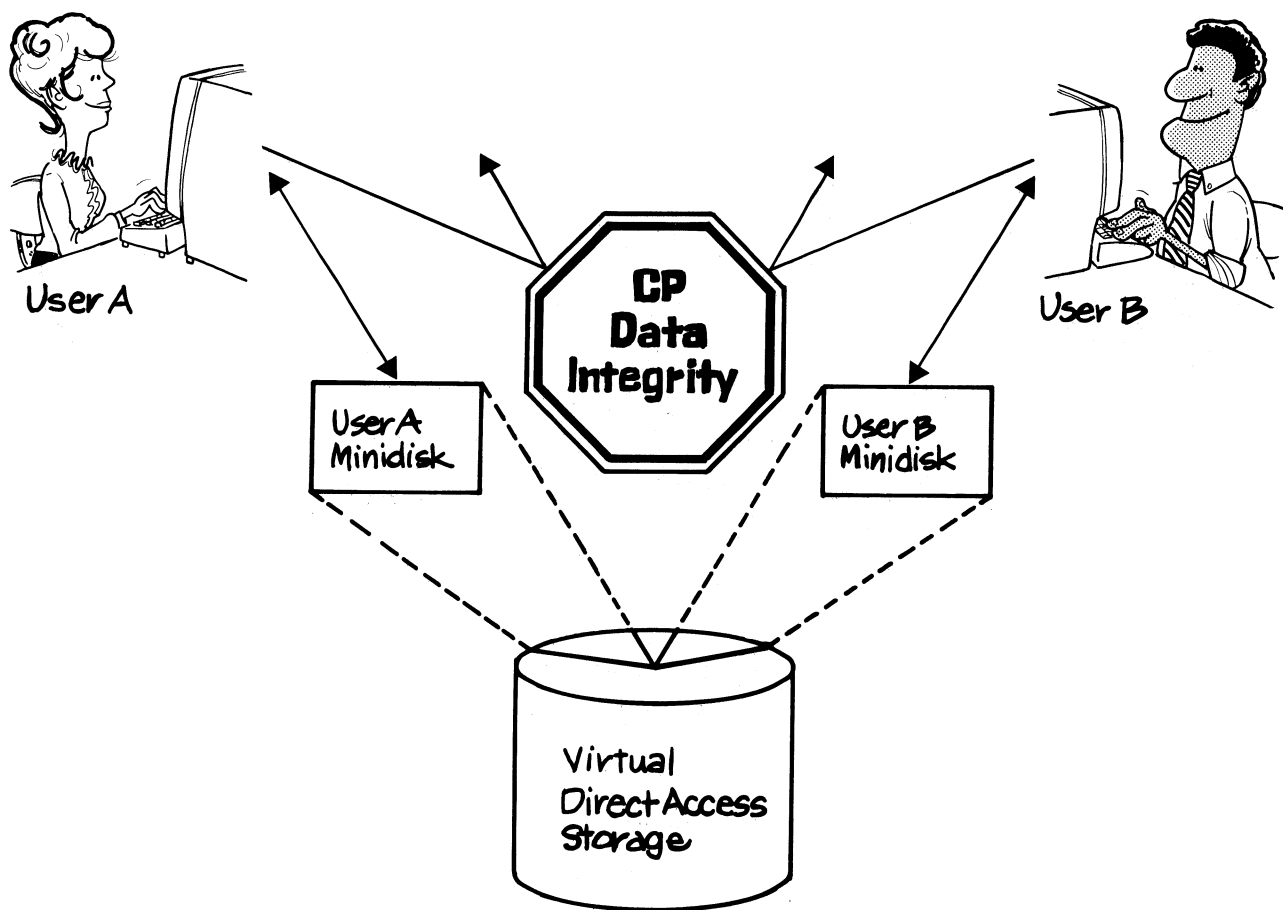


Figure 6. CP Protects Minidisks from Unauthorized Use.

⁴ FBA stands for Fixed Block Architecture. The measurement of data on FBA direct access storage devices is in a quantity of fixed-size blocks rather than cylinders, as is the case for other direct access devices.

Shared Access to Minidisks

CP assures that access to a minidisk is limited to the minidisk owner, or to other VM users who have been given access authorization by the owner (Figure 6 on page 24). Minidisks can be shared through use of a CP command (LINK). A password that authorizes minidisk sharing must be included as an operand on the MDISK directory statement that defines the shared disk. The authorized access can be for read/write or read-only access. See the LINK command in *VM/SP CP Command Reference*.

Two CP commands (DEFINE and LINK) make minidisks available to virtual machines. A CMS command (ACCESS) associates, for CMS, minidisks with the virtual machines that use them. CMS uses the filemodes (A-Z) assigned to minidisks to establish an alphabetic search order when going through a virtual machine's minidisks looking for a given file. Unless told to do otherwise, CMS first attempts to retrieve files from the user's normal work minidisk (A-disk). If unsuccessful, CMS then continues searching other minidisks accessible by that virtual machine.

Minidisk Initialization

Before data is stored on a minidisk it must be formatted to be compatible with the type of records being stored. Service programs are available to format:

- CMS disks
- CP disks
- OS, VSE, and CMS/VSAM disks.

Permanent minidisks retain their formatting across sessions, therefore, they need to be formatted only once. However, CP destroys temporary minidisks when the owner logs off, or when the user returns the temporary minidisk to CP (DETACH command). Temporary minidisks must therefore be formatted for each session they are used.

Reference

VM/SP Planning Guide and Reference, SC19-6201, describes procedures for assigning virtual machine minidisks in the VM/SP system directory.

VM/SP CMS User's Guide, SC19-6210, discusses minidisks as virtual direct access storage.

VM/SP CP Command Reference, SC19-6211, describes the format and usage of CP commands available to the general user.

VM/SP CMS Command Reference, SC19-6209, describes the format and use of CMS commands.

Spooling

Overview

- Spooling gives VM/SP users *apparent* immediate access to real unit record devices.
- CP holds spooled data on direct access storage.
- CP determines how to process spooled input or output files through the directory or from commands issued by the user.
- CP can send spooled output to another user as well as to a system device.
- The spooling process can gather a copy of virtual console data.
- The system (spool) operator manages spool direct access storage, real unit record devices, and the flow of spooled file traffic.

How CP Manages Spooling

Every virtual machine has to have access to a system's real unit record devices (reader, punch, and printer). CP provides orderly access to these devices by holding input and output files on direct access device queues while the files wait their turn to be processed. CP also manages the flow of each of these files to the appropriate unit record device when the device becomes available. This method of giving all virtual machines a fair share of unit record device usage is called spooling.

You might picture "spooling" as files placed end to end on direct access storage, as if on a spool. However, in practice spooled files are more likely to be stored in areas randomly located in spool storage.

Consider this example that points out the need for spooling. Suppose that the system printer is busy printing a large file. At that time CP receives requests from three virtual machines, each to print a file on that same system printer. CP sends each user a message saying in effect "Okay, I'll have it printed when the system printer is free." All three users are then free to work on other tasks. At this time it appears to the three users that their files are on the way to the system printer. Actually, CP has stored each file on direct access storage spool space while each file waits for its turn to go to the system printer. Figure 7 on page 27 illustrates the CP spool management facility accepting, in a parallel fashion, files to be printed and sending them, in a serial fashion, to the system printer.

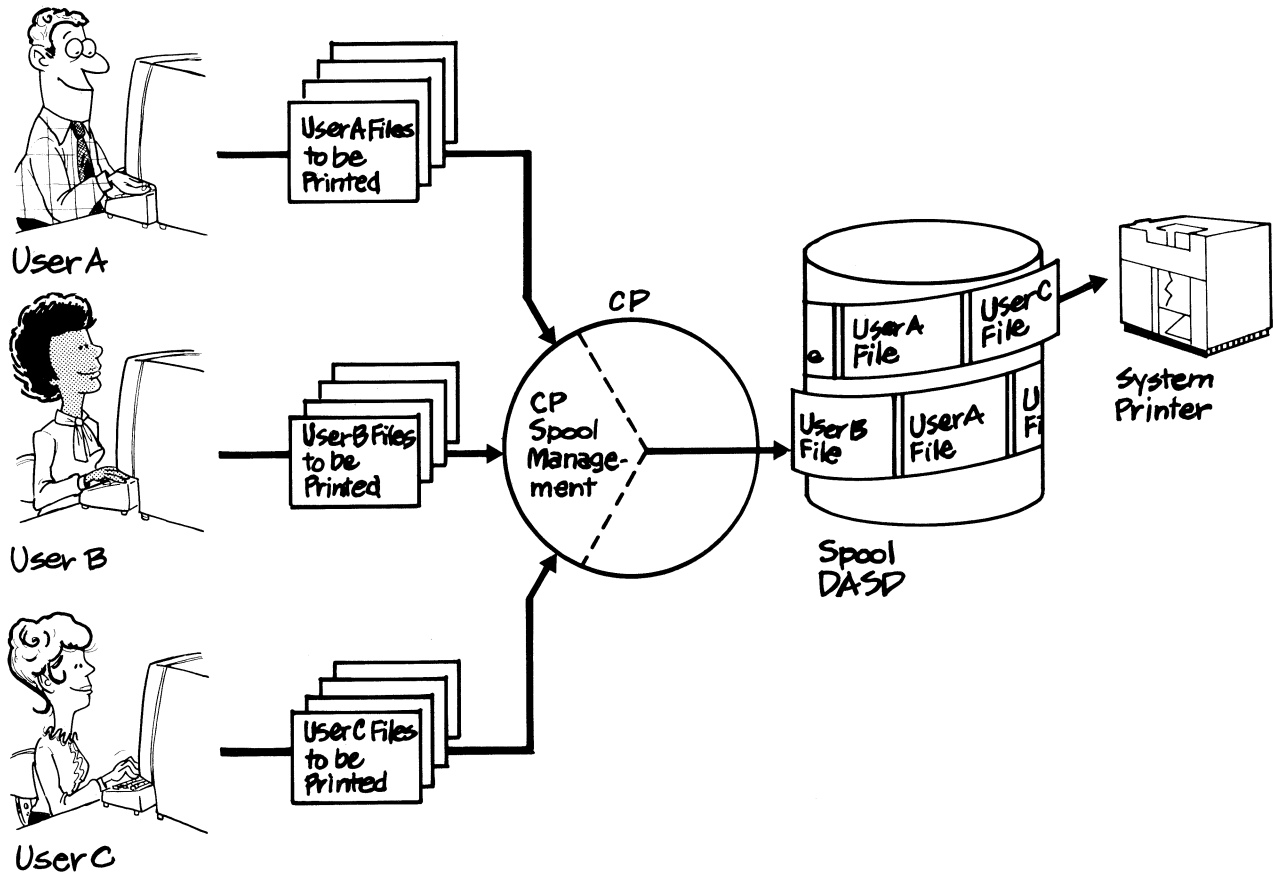


Figure 7. Shared Use of the System Printer through Spooling

The rate at which a spooled file progresses from spool storage to the system printer depends to a large degree on the priority class assigned to that file by CP compared to those priorities assigned to other printer spool files. CP gets the information it needs to assign this value either from a system directory or from a user request to CP to change the spooled file's class (CHANGE command).

To process spool files, CP must receive control information from the SPOOL command, either as issued by users or from the system directory. This control information tells CP such things as:

- Which virtual devices are being spooled
- The virtual machine class
- Routing data
- How output should be printed.

Some VM/SP Basics

After a file's spooling process is complete, CP sends messages to the originating virtual machine, indicating a successful completion or an error condition.

Spooled Virtual Punch and Virtual Reader Data

CP handles spooled output to a virtual punch much like spooled output to a virtual printer. CP accepts the punch request, responds to the user, and places the output file in spool storage. When it is that file's turn to be processed, CP directs the file to the system punch as requested by the user (SPOOL command).

Receiving virtual machine input via a virtual reader is somewhat different from printer or punch spool activity. For example, users need not be logged onto VM/SP when the spooling to their virtual readers takes place. In addition, the users (not the operator) determine when files are loaded from spool storage into their virtual machines.

Files spooled to a user's virtual reader normally come from:

- Files from a user in another virtual machine
- The system operator running a card deck in the real system reader with the file directed to this user's virtual reader.

In either case, CP sends the user a message saying that a file(s) exists in the user's virtual reader. The user can:

- Examine information about the file or the file content while the file remains in the virtual reader (RDRLIST or PEEK command)
- Load the file(s) onto the user's minidisk (RECEIVE command)
- Purge reader file(s) (DISCARD command).

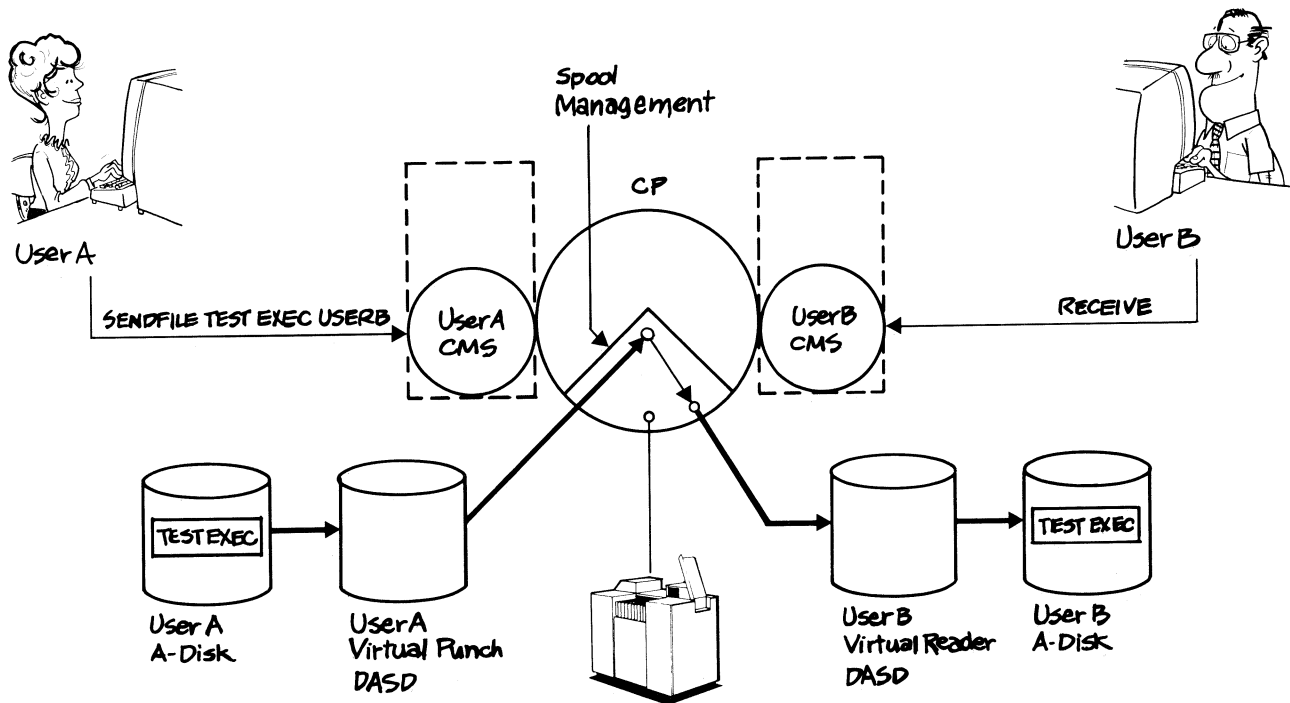


Figure 8. Spooling Virtual Punch Output to Another User

Spooling from the User's Point of View

From the CMS user's point of view, requesting unit record input and output operations is quite simple. Often it involves only a single command, such as READ, PUNCH, or PRINT. Commands such as SPOOL or CHANGE can be used to request that CP change the way the system normally processes unit record data. CP can also supply information about, or change the status of, a spooled file up to the time CP processes it on a real system device. However, CP normally completes the operation without making the user aware that spooling is taking place.

Spooling results from requests to:

- Print a file.
- Punch a file. Virtual punch spooling also occurs when a user sends a file to another virtual machine. Virtual punch spooled output is sometimes routed to the target virtual machine's reader, rather than to the real system punch (SPOOL command).
- Read a card deck from the real card reader. Virtual reader spooling also occurs when a virtual machine receives files sent (punched) from another virtual machine.

Some VM/SP Basics

- Collect virtual console data, such as commands issued from the console and system messages displayed at the console (SPOOL CONSOLE command).
- Send files or notes to other users on the system (SENDFILE command).

Under CMS, spooled virtual punch output or reader input are normally used for exchanging data among virtual machines, rather than for reading or punching cards on real system devices. For example, when the SENDFILE command is used to send a file or note to another user, both the sending user's virtual punch and the receiving user's virtual reader are involved in the process. However, no real system reader or punch device is involved. Figure 8 on page 29 shows a typical exchange of spooled data between virtual devices without involving the real system devices.

The user logging onto VM/SP becomes the operator for the virtual machine that CP creates for that user. This allows control over certain spooling functions for files associated with that virtual machine. For example, the user can tell CP:

- Where spooled output is to be sent
- How to handle input from the virtual reader
- How many copies of printed output are needed.

After releasing an output file to CP for spool processing, the user still has some control over that file. For example, the user can:

- Clear the file from spool (PURGE command)
- Change the way CP handles the file by changing the file's class (CHANGE command)
- Rearrange the order of spooled files (ORDER command)
- Transfer the file to another valid queue (TRANSFER command).

CP complies with this type of request if it receives the command before actual processing by the unit record device begins.

Spooling from the Operator's Point of View

A system operator assigned to manage spool operation (a spool operator) controls movement of files between spool storage and real unit record devices. The operator alone has authority to issue commands related to spooling that affect real system devices. This authority, beyond that of a general user, includes issuing commands that affect spooled files. For example, the operator can:

- Restart processing of an output file from the beginning (BACKSPAC command)

- Start and stop real spool devices (START and DRAIN commands)
- Clear a file being processed (FLUSH command)
- Put files on, or remove files from, system hold status (HOLD and FREE commands)
- Change the system printer's type style (LOADBUF command)
- Control the number of copies and the line spacing of printed files (REPEAT and SPACE commands)
- Transfer spool files to and from tape storage (SPTAPE command).

It is the spool operator's responsibility to:

- Make sure that the real unit record devices are ready to satisfy user input/output requests. For example, the operator might need to put a specific print train on the printer, or load a particular type of printer paper requested by a user.
- Handle output. For example, the operator must separate printed output by user.
- Manage spooling priority. For example, although several users may be assigned equal priority, one user might send a message to the operator asking that a file be printed as soon as possible. The spool operator can cause that user's output to be run ahead of the others, even though it did not arrive first on the spool. Also, the spool operator can elect to send all small files to the printer and save the large time-consuming files for a less-busy time.
- Temporarily relocate some spooled files from direct access to a tape device if CP sends a message that the direct access spool area is almost full.

How to Retrieve Virtual Console Data through Spooling

As a virtual machine operator the user can save a copy of virtual console input and output in a spool file. There are times when this can be very useful. For example:

- When a procedure is run from which displayed data or a record of the procedure itself must be saved.
- When a virtual machine is run in a disconnected state, perhaps to allow the user to temporarily leave the area or to operate from a different virtual machine. This allows the recording of messages that would be displayed at the terminal if the virtual machine had remained connected to VM/SP.

Console spooling can be started, interrupted, or stopped by issuing appropriate options with the SPOOL CONSOLE command.

Some VM/SP Basics

A Summary of CP's Spooling Activity

In summary, to support VM/SP spooling activity, CP:

- Provides virtual unit record devices, using direct access storage devices as intermediate storage for data passing between virtual machines and the real unit record devices
- Manages use of the system's real unit record devices
- Provides for data exchange among the users' virtual machines, spool storage, and input/output devices
- Handles commands and responses related to spooling.

Reference

VM/SP Planning Guide and Reference, SC19-6201, describes system directory entries that affect spooling.

VM/SP Operator's Guide, SC19-6202, describes in more detail spooling from the spool operator's point of view.

VM/SP CP for System Programming, SC24-5285, describes in more detail spooling from the system programmer's point of view.

VM/SP CMS User's Guide, SC19-6210, describes spooling support for virtual machines running CMS.

VM Running Guest Operating Systems, SC19-6212, describes spooling for operating systems other than CMS running in a virtual machine.

VM/SP Is a Conversational System

Overview

- Users communicate to VM/SP through commands or responses to prompting messages.
- VM/SP communicates to users through system messages that:
 - Give a warning or indicate error conditions
 - Provide information or system status
 - Ask for a response.
- Only the system operator issues commands that affect real system resources.

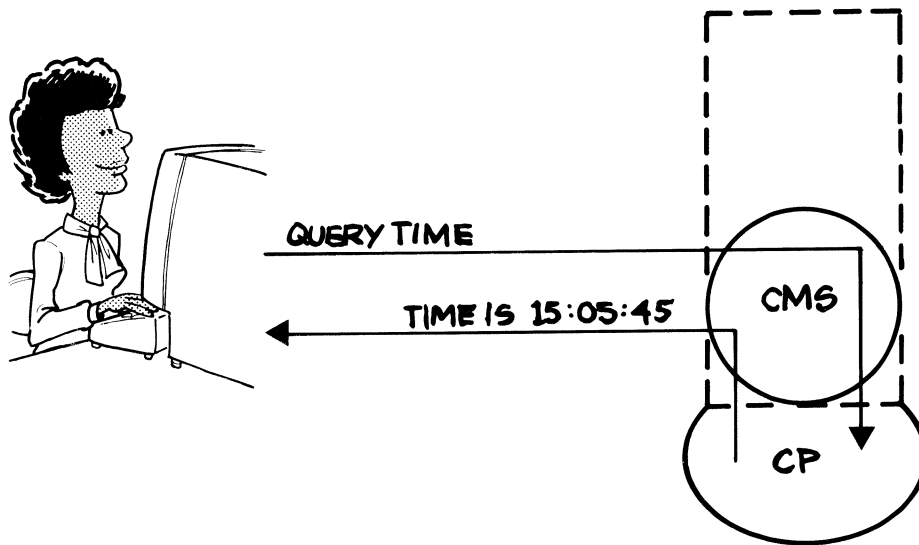


Figure 9. CP Commands May Be Issued to CMS

Some VM/SP Basics

How a General User Communicates to VM/SP

General users communicate to VM/SP using:

- Commands to CP that relate to virtual resource management
- Commands to CMS that relate to work flow within a virtual machine.

The most common communication from a user to CP is the logon procedure to begin a VM/SP session, or the logoff procedure to end a session. Other examples include the loading of an operating system into a virtual machine, or requesting system status information. In each of these cases the user communicates directly to CP.

So far this book has talked mostly about communication between a user and CP. This is because CP is involved in managing a user's virtual machine session right from the time the user logs onto VM/SP. On the other hand, communication between a user and CMS takes place only after CMS is initial program loaded (IPLed) and in control of the user's virtual machine.

Users can continue to communicate with CP after their virtual machines are under control of CMS. For example, CMS users might need to have CP assign additional direct access storage work space to their virtual machines or request the correct time. They can simply key in the appropriate CP command without ever leaving the CMS environment (Figure 9 on page 33).

During a session when CMS controls the virtual machine, the user communicates more often with CMS than with CP. This is because, after CP has supplied a virtual machine for the user, most session activity usually relates to CMS-controlled work within the virtual machine.

VM/SP provides a set of commands that communicate with CMS only. CMS commands can be issued only when CMS is controlling a virtual machine. CP commands can be issued within either the CP or the CMS environment.

There is more about communication between the user and CMS later in this book under "Communicating with CMS" on page 99.

System Messages Communicate to Users

VM/SP has an extensive set of messages for communicating to users. CP and CMS have their own messages. The identifier for each message tells a lot about the message. For example, the message identifier DMSGEN111E tells a user that CMS issued this message (DMS), the module in CMS that generated this message (GEN), the message number (111), and that this message indicates an error condition (E).

Users can request, via the CP SET command, that only specified portions (or none) of certain messages actually display at their terminals. For example,

```
SET EMSG TEXT
```

requests that only the text portion (no message identifiers) of error messages be displayed.

How the System Operator Communicates to VM/SP

The system operator's primary job is to manage CP's control of real system resources and the virtual machine environment. To handle these tasks the operator is authorized to use CP commands not available to general users. A privilege class entry in the system directory for the operator's virtual machine gives the operator authority to issue these privileged commands. Using these commands, the operator can:

- Start and stop real devices
- Control system functions
- Provide system-level management of user virtual machines.

CP's communication to the system operator is usually in the form of system status messages. The system operator's communication to CP might include commands that affect system resource management. For example:

- QUERY - to request system status
- ORDER - to change the relative positions of spooled files
- SHUTDOWN - to stop VM/SP system activity.

Reference

VM/SP Operator's Guide, SC19-6202, describes commands whose usage is restricted primarily to the system operator.

VM/SP CMS Command Reference, SC19-6209, describes commands for communicating with CMS.

VM/SP CP Command Reference, SC19-6211, describes commands that can be used by any user to communicate with CP.

VM/SP System Messages and Codes, SC19-6204, describes system message format and identifier meanings, and explains the message text plus the expected response for all messages issued by VM/SP.

Some VM/SP Basics

A Summary of VM/SP Concepts

- A single real VM/SP system can give many users their own individual virtual machines. CP manages real system resources and provides the virtual machines.
- Each virtual machine is a functional equivalent of a real system. Each virtual machine shares use of real processor function, storage, console, and input/output device resources.
- CMS is VM/SP's conversational operating system for controlling work flow in a virtual machine.
- When a user logs on to VM/SP, CP creates a virtual machine as defined in the system directory.
- Any one of several operating systems other than CMS may be used to control work flow in a virtual machine.
- Paging makes real storage usage more efficient by moving inactive 4K-byte blocks of virtual storage to direct access storage.
- Spooling allows many users to share unit record devices by holding input and output data on direct access storage until the needed real device is available to process it. CP implements the spooling facility.
- VM/SP is a conversational system in that it communicates to users through system messages and prompts, and receives communication from users through responses and commands.

Chapter 2. Putting VM/SP to Work

Using VM/SP means different things to different people. For example, a programmer, a technical writer, and a production analyst use VM/SP in entirely different ways. But broadly speaking, the most common work that people do on VM/SP can be classified as:

- Executing programs
- Developing programs
- Creating information files
- Retrieving information files
- Running other operating systems.

This chapter discusses a variety of jobs that VM/SP makes easier to do. Remember, the purpose of this chapter is to make you aware of the kinds of things that can be done on VM/SP. It is not to be used as a guide for actually doing these things. Other books of the VM/SP library contain guide and reference information for using VM/SP. Each topic includes references for finding additional information.

Chapter Contents

Basics about Working in the Virtual Environment	39
CP Commands	39
Communicating with Other Virtual Machines	40
Communicating with the System Operator	41
Virtual Machine Input and Output	41
Developing Application Programs under CMS	44
Steps in the Application Program Development Process	44
Using the VM/SP System Product Editor for Program Development	45
Using CMS to Assemble Programs	45
Using CMS to Compile High-level Language Programs	45
Debugging Application Programs	50
Checking a New Program	51
CMS Facilities to Help Debug Programs	51
CP Facilities for Debugging Programs	52
Handling Problems with VM/SP IPCS	54
How IPCS Works	54
Executing Programs under CMS	57
Preparing for Program Execution	57
A Command Recap	59
Processing Programs with the RUN Command	59
Processing Jobs in Batch Mode under CMS	61
What the CMS Batch Facility Is	61
Using the CMS Batch Facility	61
What the VM Batch Subsystem Is	63
Using the VM Batch Subsystem	63
Running Operating Systems in a Virtual Machine	65
Some Other Programs That Run under VM/SP	67

Basics about Working in the Virtual Environment

Overview

Most user sessions on VM/SP have some things in common. Among these are:

- Logon, IPL (loading an operating system), and logoff
- Other communication to CP, such as requesting system status or requesting changes to a virtual machine
- Communication to CMS (or some other operating system) to manage work in a virtual machine
- Communication with the system operator
- Communication with other users (virtual machines)
- Handling virtual machine input and output.

CP Commands

There are two types of commands for communicating to CP. The first type consists of privileged usage (other than general user) commands for managing the real system processor or devices, or those that affect the overall virtual machine environment. Examples include commands such as VARY (make a device available or unavailable to CP), SHUTDOWN (terminate VM/SP functions), and FORCE (force logoff of a user). A system directory entry authorizes the system operator to issue privileged commands. If a general user attempts to issue one of these commands, CP sends a response indicating rejection of the request. The *VM/SP CP Command Reference* fully describes system operator tasks and the commands available to do these tasks.

A second type of CP command consists of those used by general users during a VM/SP session. These commands request CP to do such things as:

- Display status information about the system or about a virtual machine. For example:
 - How busy is the system processor, or what is the current paging activity?
 - Has a file been printed yet?
 - Are there files in the virtual reader?

Putting VM/SP to Work

- Make changes to a virtual machine. For example:
 - Change the way CP is to handle spool files. CP normally uses internal (default) processing specifications or system directory entries to determine how spooled files are to be processed. For example, the number of printed output copies a user is to receive can be changed.
 - Change a virtual machine configuration. For example, the size of a user's virtual storage can be increased or decreased.
 - Remove a device from a virtual machine configuration. For example, suppose that a virtual machine has had temporary use of a magnetic tape device. When the device is no longer needed, the user must detach it from the virtual machine so that it will be available to other users.

The *VM/SP CP Command Reference* describes CP commands that are available to most users.

CP commands can be issued from either the CP or CMS environment. From the time a user turns on a terminal until the CMS operating system is loaded, the user works directly with CP. During this time the system does not recognize CMS commands. When the CMS IPL process is complete, the virtual machine enters the CMS environment. While a virtual machine is running under CMS, the user can continue to issue CP commands.

Figure 10 on page 41 illustrates typical communication to CP. The system operator requests CP to add a magnetic tape device to a user's virtual machine configuration. The user then requests CP to send a message confirming that the tape device has been made available to him.

Communicating with Other Virtual Machines

VM/SP users can exchange messages and data files with other users in the same system. If a data file is sent to another user, VM/SP transfers the file to that user's virtual reader. If a message is sent to another user, VM/SP displays that message on the receiving user's console if that user is logged on. Otherwise, VM/SP displays a message at the sender's console indicating that the receiving user is not logged on. A user can request that CP not interrupt with messages (SET MESSAGE OFF). In that case CP does not send messages to that user, and informs the message sender that the receiving user has turned off message reception.

CP's special message (MSG) facility can be used to send messages to other virtual machines programmed to accept these messages. The message text is received by a program in the receiving virtual machine rather than being displayed on the receiving virtual machine's console. The SET MSG and the AUTHORIZE commands allow the receiving user to accept or reject incoming special messages.

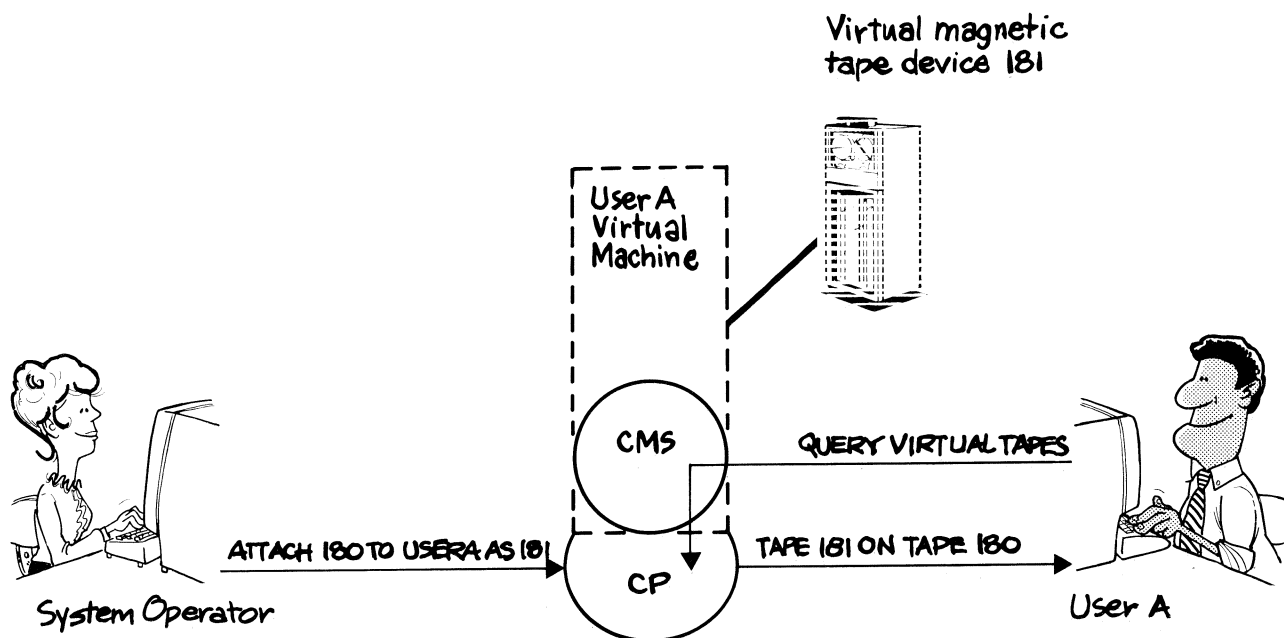


Figure 10. Communicating with CP

CP also supports facilities for data exchange with users on other systems. This topic is covered later in this book under “RSCS Networking” on page 141.

Communicating with the System Operator

There are times when program execution in a virtual machine depends on some action being taken on a real device. For example, a magnetic tape must be mounted on a tape device, or printed output must be handled in some special way. Situations like these require communication to the system operator. Except for lengthy or complex requests, the user can simply send the operator a message (MESSAGE command with the OPERATOR operand). The operator’s response may be a request for more information, or it may say that the request has been handled.

Virtual Machine Input and Output

The terms “virtual reader” and “virtual punch” do not necessarily imply any data exchange between a virtual machine and a real reader or punch device. In general, “virtual reader” refers to VM/SP’s facility for supplying input to a virtual machine. “Virtual punch” refers to VM/SP’s facility for distributing virtual machine output. Neither of these terms refers specifically to a real system device. A virtual reader or virtual punch may be used for data exchange between a virtual machine and the real system reader or punch device. On the other hand, a virtual reader may receive data sent by a user in another virtual machine. Likewise, a virtual punch may serve as a vehicle for sending data to another virtual machine.

Some examples of virtual machine input and output data include (Figure 11 on page 43):

- Virtual reader
 - Input card decks loaded by the system operator into a real card reader and read into a virtual reader
 - Files sent by users in other virtual machines.
- Virtual punch
 - Output files to be punched on a real system card punch
 - Files sent to a user on another virtual machine.
- Virtual printer
 - User-created files to be printed on the system printer
 - Files resulting from diagnostic dumps
 - Spooled virtual console output.

Options of the SPOOL command tell CP how to handle spooled virtual reader, punch, printer, or console data. For example, virtual printer output or spooled console data is normally sent to the real system printer for processing. However, the SPOOL command can be used to route these outputs to another user instead of to the system printer.

CP displays a system message at a user's terminal to advise if files exist in the user's virtual reader. This may happen either at logon (if files were sent while the user was logged off), or during a session (at the time CP puts files into the virtual reader). The user can request CP to display information about the virtual reader files (QUERY READER ALL). The user can then transfer the files from the virtual reader to virtual direct access storage (RECEIVE command). Or, the user can clear them, either selectively or all at once, from the virtual reader (PURGE command).

It is possible to destroy old files on virtual disk storage when loading new ones from the virtual reader. This happens when the file in the reader has the same identifiers as a file on the minidisk to which the reader file is being transferred. The RECEIVE command displays a warning message to advise the user that another file with the same identifiers exists. However, some procedures, such as DISK LOAD, for loading reader files can overlay files on virtual disk without warning. Facilities are available for checking identifiers of files in the virtual reader and on virtual direct access storage before loading reader files.

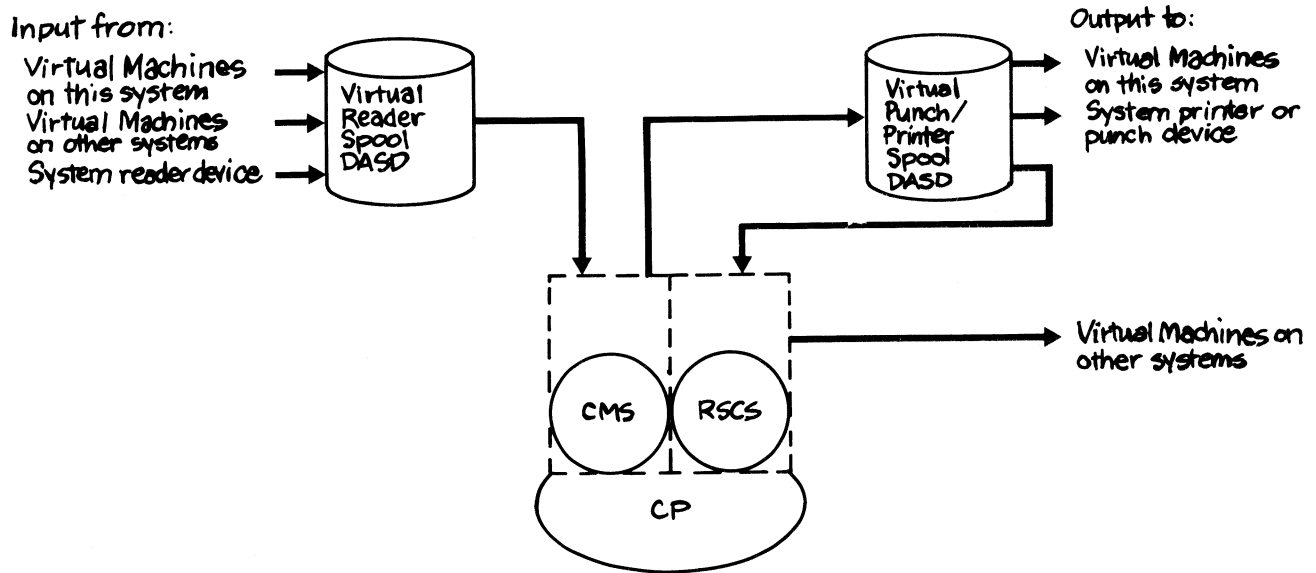


Figure 11. Virtual Machine Input and Output.

CMS users can use the SENDFILE command to send files or notes to other users on their system or other systems. For example,

```
SENDFILE PARTS AUTO ROCKY
```

sends the file named PARTS AUTO to the virtual reader of a user identified as ROCKY in a special "names" file that the user has created using the NAMES command.

The RDRLIST command under CMS displays information about files in a virtual reader.

Reference

VM/SP CMS Primer, SC24-5236, tells how to log onto and off from VM/SP, and guides the user through many CMS functions of creating and editing CMS files.

VM/SP CP Command Reference, SC19-6211, contains descriptions of all commands used to communicate with CP.

VM/SP CMS Command Reference, SC19-6209, describes all commands that can be issued to CMS for managing a virtual machine.

Developing Application Programs under CMS

Overview

To help with program development, VM/SP provides:

- The VM/SP System Product Editor for writing and editing source code
- A VM/370 assembler
- Support for a variety of program compilers, such as FORTRAN, COBOL, and PL/I.

A few definitions at this point might be helpful for readers not familiar with developing programs.

- Source program — a program expressed in a source language, such as assembler, COBOL, or FORTRAN.
- Object program — the program that results from using an assembler or compiler to translate source program code to machine code (code that is executable on a computer)
- Assemble — the process of converting a source program written in assembler language to an object program in machine language.
- Compile — the process of converting a source program written in a high-level language, such as COBOL or PL/I, to an object program in machine language.
- Text file — a CMS file to which an object program is written during an assembly or compile process.
- Listing file — a CMS file to which is written a listing of source resulting from an assembly or compile process.

Steps in the Application Program Development Process

The steps in the application program development process are:

1. Write a source code program in assembler or high-level language
2. Assemble or compile the source code
3. Correct assembly or compile-source-code errors
4. Load the object program

5. Identify files to be processed by the program
6. Execute the object program
7. Correct any program errors
8. Execute the object program in production mode.

Using the VM/SP System Product Editor for Program Development

The System Product Editor is a VM/SP facility for creating and editing CMS files. This is described under “VM/SP System Product Editor” on page 110. Manuals listed at the end of this topic provide information about using the System Product Editor.

Using CMS to Assemble Programs

The CMS assembler (ASSEMBLER XF) program converts program source files written in assembler language code into object programs in machine language. Source files to be assembled must have a filetype of ASSEMBLE. The assembly process produces two output files on direct access storage. Each of these output files has the same filename as the source file. However, one has a filetype of TEXT, and the other a filetype of LISTING. The resulting program is ready to be run in a virtual machine.

The assembly process is initiated using the ASSEMBLE command under CMS. For example, issuing the command:

```
ASSEMBLE PROGA
```

processes an assembler source file named PROGA. Options can be included on the command line that affect the assemble process. The filetype, although not specified, is assumed to be ASSEMBLE.

CMS locates the source file named in the command. It then invokes the assembler program to run in the user’s virtual machine. It writes output TEXT and LISTING files to the disk from which it obtained the source file, if possible. Otherwise, it writes the two output files to the first available read/write disk in the user’s access order.

Using CMS to Compile High-level Language Programs

Among the high-level program compilers supported by VM/SP are:

- IBM BASIC
- OS/VS COBOL and DOS/VS COBOL
- OS FORTRAN and VS FORTRAN
- OS PL/I and DOS PL/I
- Pascal/VS.

Putting VM/SP to Work

Unlike the ASSEMBLER XF program, high-level language compilers are not a part of VM/SP. These licensed programs such as COBOL, FORTRAN, PL/I, and APL can be ordered separately.

Any references to language licensed programs in this manual are for information only and are not intended to be used as a guide for using the programs. Each of these programs has its own information library. Anyone intending to create or use programs using any of these high-level languages should get a copy of each of the manuals that support that language compiler. References to some of these manuals are at the end of this topic.

Compiling a program written in one of the supported high-level languages is similar to processing an assembler language program. For example:

```
PLIOPT MYSOURCE
```

invokes the PL/I optimizing compiler to process a PL/I source file named MYSOURCE. Options that affect the compile process can be included on the command line. Like the assemble process, the compiler writes a TEXT file and a LISTING file to disk.

Figure 12 on page 47 illustrates the invoking of a PL/I compiler to process a source PL/I program (MYSOURCE). The top part of the figure shows the compiler and the source program being loaded into the user's virtual machine. The lower part of the figure shows the output files produced by the process.

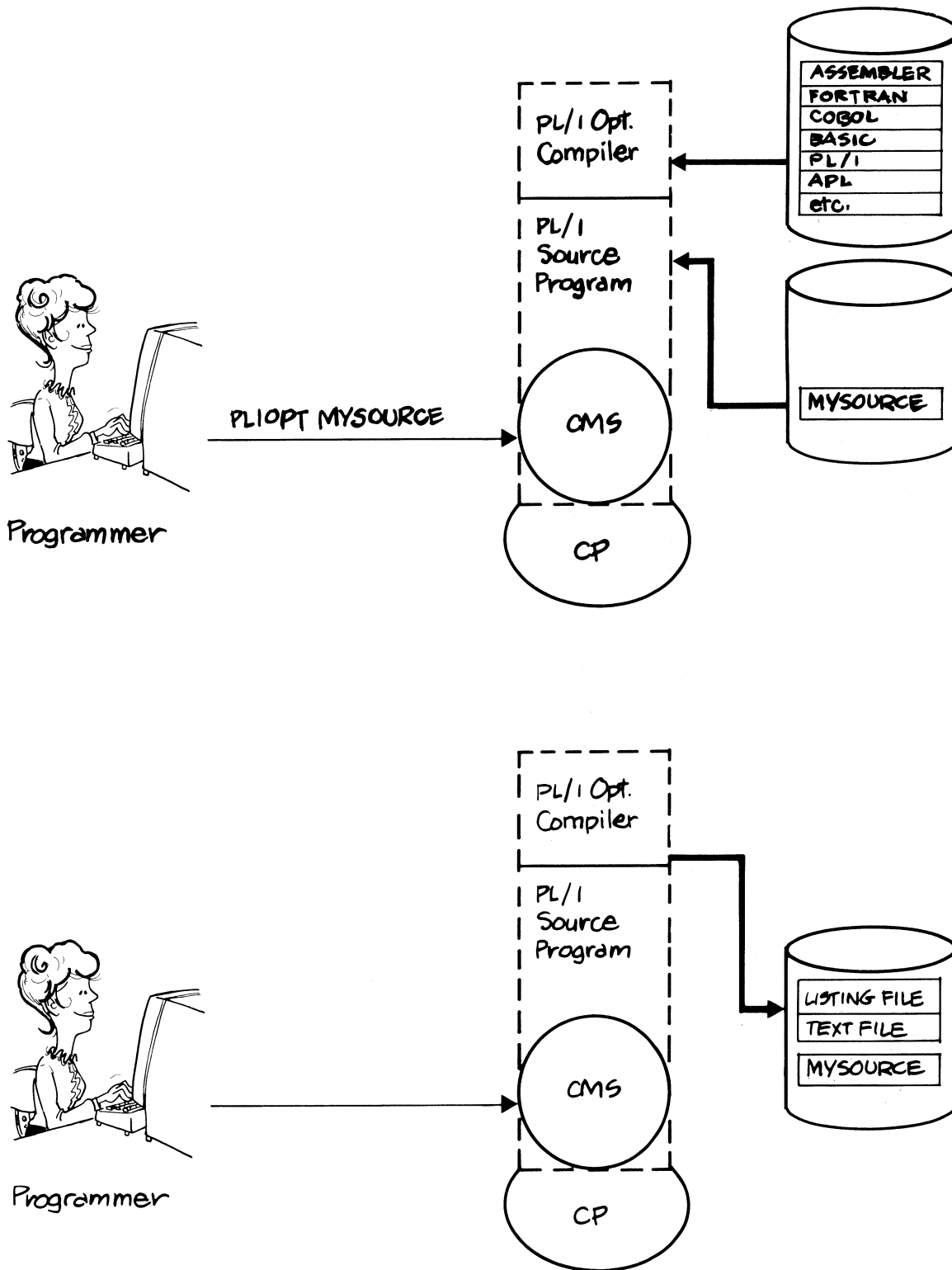


Figure 12. Compiling a High-Level Language Program.

Putting VM/SP to Work

Reference

VM/SP Application Development Guide, SC24-5247, tells how to compile, link, load, run, test, and debug FORTRAN and COBOL programs using CMS.

VM/SP CMS Macros and Functions Reference, SC24-5284, provides additional information about how to write and process program source code using CMS.

VM/SP CMS for System Programming, SC24-5286, provides information about how to develop programs under CMS for OS and VSE.

VM/SP CMS Primer, SC24-5236, describes the basics of creating and editing source files.

VM/SP CMS Primer for Line-Oriented Terminals, SC24-5242, describes creating and editing source files for VM/SP users of line-oriented (line mode) video display terminals.

VM/SP CMS Command Reference, SC19-6209, describes the ASSEMBLE command and its options.

The following lists some of the information available for the programming languages supported by VM/SP. See *IBM System/370 and 4300 Processors Bibliography*, GC20-0001, for additional information.

1. ASSEMBLER XF

- *OS/VS, VM/370 Assembler Programmer's Guide*, GC33-4021.

2. COBOL

- OS/VS COBOL Compiler and Library (program number 5740-CB1)

General Information, GC28-6470

CMS User's Guide for COBOL, SC28-6469.

- OS COBOL Interactive Debug (program number 5734-CB4)

User's Guide and Reference, SC28-6465.

- DOS COBOL Compiler and Library (program number 5736-CB1)

Programmer's Guide, SC28-6441.

3. FORTRAN

- VS FORTRAN Compiler and Library (program number 5748-FO3)

General Information, GC26-3983

Application Programming Guide, SC26-3985.

- FORTRAN Interactive Debug (program number 5734-F05)

Guide and Reference, SC28-6885.

4. PL/I

- OS PL/I Optimizing Compiler and Libraries (program number 5734-PL3)

General Information, GC33-0001

CMS User's Guide, SC33-0037.

- OS PL/I Checkout Compiler (program number 5734-PL2)

General Information, GC33-0003

CMS Reference Guide, SC33-0047.

- DOS PL/I Optimizing Compiler (program number 5736-PL1)

Programmer's Guide, SC33-0008

CMS User's Guide, SC33-0051.

5. RPG - DOS/VS RPG II (program number 5746-RG1)

General Information, GC33-6120

User's Guide, SC33-6074.

6. APL - VS APL (program number 5748-AP1)

General Information, GH20-9064

CMS Terminal User's Guide, SH20-9067.

7. BASIC - IBM BASIC (program number 5668-996)

General Information, GC26-4023

Application Programming Guide, SC26-4027.

8. PASCAL - Pascal/VS (program number 5796-PNQ)

Notice, G320-6387.

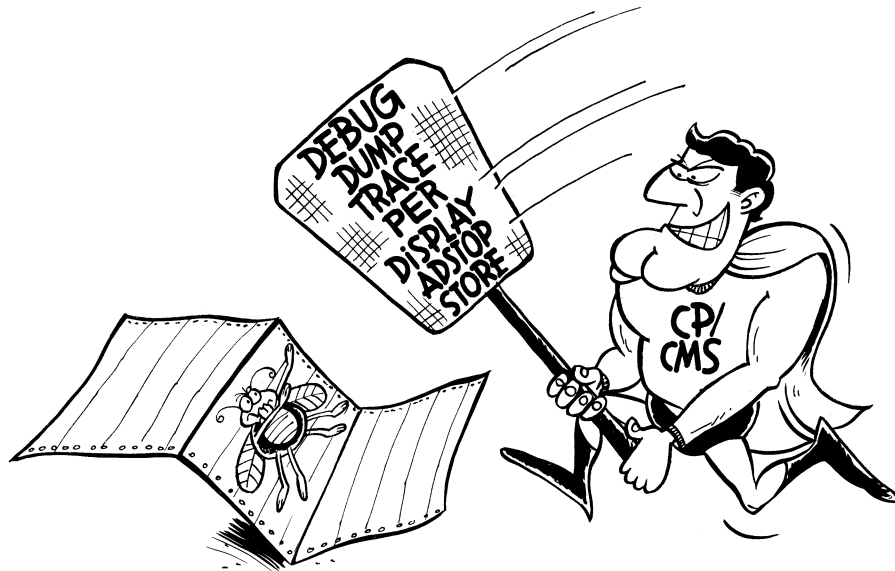
Putting VM/SP to Work

Debugging Application Programs

Overview

VM/SP has a variety of facilities to help find problems in application programs. These include:

- Program listing files (filetype LISTING) resulting from an assemble or compile process.
- The DEBUG command (CMS)
- The SVCTRACE command (CMS)
- The TRACE command (CP)
- The PER command (CP)
- DISPLAY, ADSTOP, and STORE commands (CP)



The subject of application program debugging is far too broad and complex to be described fully in this book. The purpose of this topic is to present an idea of what facilities are in VM/SP to help debug application programs, and where to find more complete information about these facilities. The *VM/SP Diagnosis Guide* provides a more in-depth discussion of debugging and executing application programs under VM/SP. A topic later in this book, “Diagnosing System Programming Problems” on page 95 discusses system program debugging.

Both CP and CMS offer facilities for debugging programs. CMS DEBUG subcommands cannot be used under CP. CP debugging commands can be issued from the CMS DEBUG environment.

This book does not discuss tools and commands for debugging programs that run under GCS. The *VM Diagnosis Guide* has that information.

Checking a New Program

A newly assembled or compiled program can be tested by loading it into a virtual machine and then starting it. For example, when a program named TEST1 is compiled, one of the results is an executable file named TEST1 TEXT. The program can be executed by issuing:

```
LOAD TEST1
START
```

Additional commands may be required in preparation for program execution. For example, if the program TEXT file resides in a system library, CMS may need to be told where to find it (GLOBAL command). Or, if the program refers to input or output files, they may need to be defined before the program is run (FILEDEF command).

CMS Facilities to Help Debug Programs

LISTING File

To help the user find application program problems, CMS provides a LISTING (filetype LISTING) file on direct access storage when a source program is assembled or compiled. Using a printout of this file, or displaying the file at a terminal, can help the user follow step-by-step program execution.

DEBUG Command

The CMS debugging aid interactively assists the user in finding and fixing program errors that:

- Produce inaccurate results
- Cause abnormal termination of processing.

A virtual machine is put in program debugging mode by simply issuing a CMS command (DEBUG). The user can then issue any of the DEBUG subcommands. Using these subcommands a user can:

- Stop or resume program execution
- Display or change storage contents
- Display or change register contents

Putting VM/SP to Work

- Dump to the virtual printer:
 - Selected portions of virtual storage
 - All of virtual storage.

SVCTRACE Command

The SVCTRACE command under CMS gives more complete data about SVC execution in a program than the CP TRACE command discussed next in this topic. It provides such information as:

- Register contents before and after the SVC
- The name of the called routine
- The location from which the routine was called
- The contents of the passed parameter list.

CP Facilities for Debugging Programs

TRACE Command

The TRACE command under CP monitors several kinds of program activity, including:

- Instructions
- Branches
- Interrupts
- I/O.

Options that affect execution can be included on the TRACE command line. For example:

```
TRACE PROGRAM BRANCH
```

traces virtual machine program interrupts and successful branches and sends the results to the virtual console.

PER Command

The program event recording (PER) command monitors virtual machine events as they occur during program execution. It does this by suspending program execution after each event. The PER command traces:

- Successful branches
- Instruction fetching and execution
- Instructions that alter general purpose registers
- Instructions that alter storage.

Other CP Debugging Commands

Useful CP debugging aids include:

- DISPLAY — to look at the contents of virtual storage, registers, or system status words (CSW, CAW, PSW).
- ADSTOP — to stop program execution at a specified instruction address before execution.
- STORE — change the contents of a storage position, register, or control word.

Reference

VM/SP CMS Command Reference, SC19-6209, describes in detail the DEBUG command and related subcommands.

VM/SP CP Command Reference, SC19-6211, describes in detail CP commands that can be used for program debugging.

VM Diagnosis Guide, LY24-5241, explains the GCS debugging facilities.

Handling Problems with VM/SP IPCS

Overview

- **IPCS is included as part of VM/SP.**
- **IPCS helps users handle system software failures.**
- **IPCS runs in a CMS virtual machine.**
- **IPCS provides three major functions:**
 - **Problem reporting**
 - **Problem diagnosis**
 - **Problem and data management.**

VM/SP Interactive Problem Control System (IPCS) is an integral part of VM/SP. It runs as an application under CMS in a virtual machine. IPCS has a number of facilities that help IBM and customer service personnel handle system software problems in an orderly, efficient, and well-recorded manner. IPCS helps prevent such time-consuming things as lengthy diagnosis of different occurrences of the same problem. It simplifies the tasks of locating existing fixes or reporting new problems to IBM.

How IPCS Works

IPCS facilities include:

- A set of commands for diagnosing and reporting problems
- Messages
- Data files for each problem, including:
 - Problem report
 - Summary of problem symptoms
 - Problem status
 - Problem data record
 - IPCS dump output
 - IPCS load map

- Customer profile.

IPCS uses information from CP ABEND dumps, CMS dumps, GCS dumps, TSAF dumps, PVM dumps, RSCS dumps, and any other dumps created by the CP command VMDUMP, Stand-Alone Dump, or DIAGNOSE X'94'. Using VMDUMP, the operator can make available to IPCS a dump of user-detected software problems. Use of IPCS to produce a dump listing or data to be used for getting help from an IBM support center is shown in Figure 13.

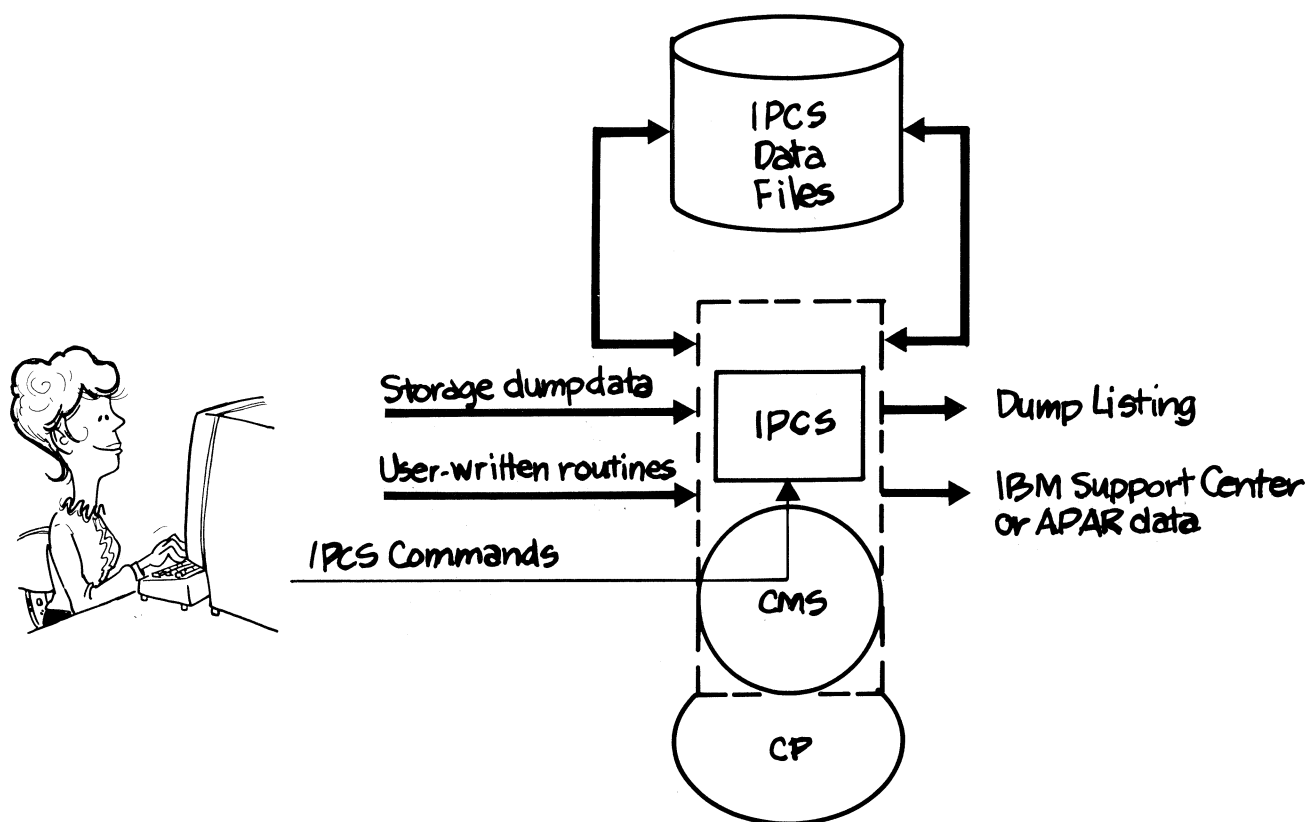


Figure 13. IPCS Is an Organized Approach to Handling Software Problems.

Dump data and IPCS commands are used to:

- Gather information needed when contacting an IBM support center for a possible fix
- Interactively examine the dump contents for possible solution
- Create an APAR,⁵ if necessary, to submit to IBM for help

⁵ An APAR is an Authorized Program Analysis Report. It is used to report program problems to IBM personnel responsible for resolving the problems.

Putting VM/SP to Work

- Update problem files in the IPCS data base.

After a problem is fixed, a complete record of the problem becomes a part of the IPCS data base to be used for diagnosing future problems.

IPCS enables users to use their own routines to:

- Scan dumps
- Format portions of dumps
- Create their own problem reports
- Create maps to append to dumps.

Reference

VM Diagnosis Guide, LY24-5241, fully describes the program and how to use it.

Executing Programs under CMS

Overview

- An application program becomes executable (a TEXT file) as a result of assembling or compiling the source code.
- In a production environment one or more debugged TEXT files can be converted into an executable MODULE file residing on a CMS disk.
- Before program execution, any input or output files referred to by the program may need to be defined.
- If the program or any data required for execution resides in a system library, that library must be identified to CMS before program execution.
- Execution is initiated by either:
 - LOADING and STARTING a program TEXT file (filetype of TEXT), or
 - Invoking a MODULE file (filetype of MODULE) by name.
- Program execution can be initiated from EXEC procedures.

As discussed in the previous topic, source code for debugging is executed by loading and executing program TEXT files that result from the assemble or compile process. However, the aim of program development is to run programs in a production environment. When TEXT files are fully debugged, one or more of them may be converted into an executable MODULE file. The module, made up of tested files, is then executed by issuing the module name from the virtual console or as a statement in an executable procedure (EXEC).

Preparing for Program Execution

Initiating the execution of an application program may require only that the name of the program module be issued. However, a few preliminary commands are normally issued to prepare for program execution. The most efficient way to do this is from an EXEC procedure, especially if the same sequence of commands is to be issued on a regular basis. That way, by issuing only the name of that procedure, the complete process, from preparation to program execution, is initiated.

If a program refers to input or output files, the FILEDEF command can be used to override certain default definition values assigned by CMS. For

Putting VM/SP to Work

example, this might apply to such things as record length or block size values for these files.

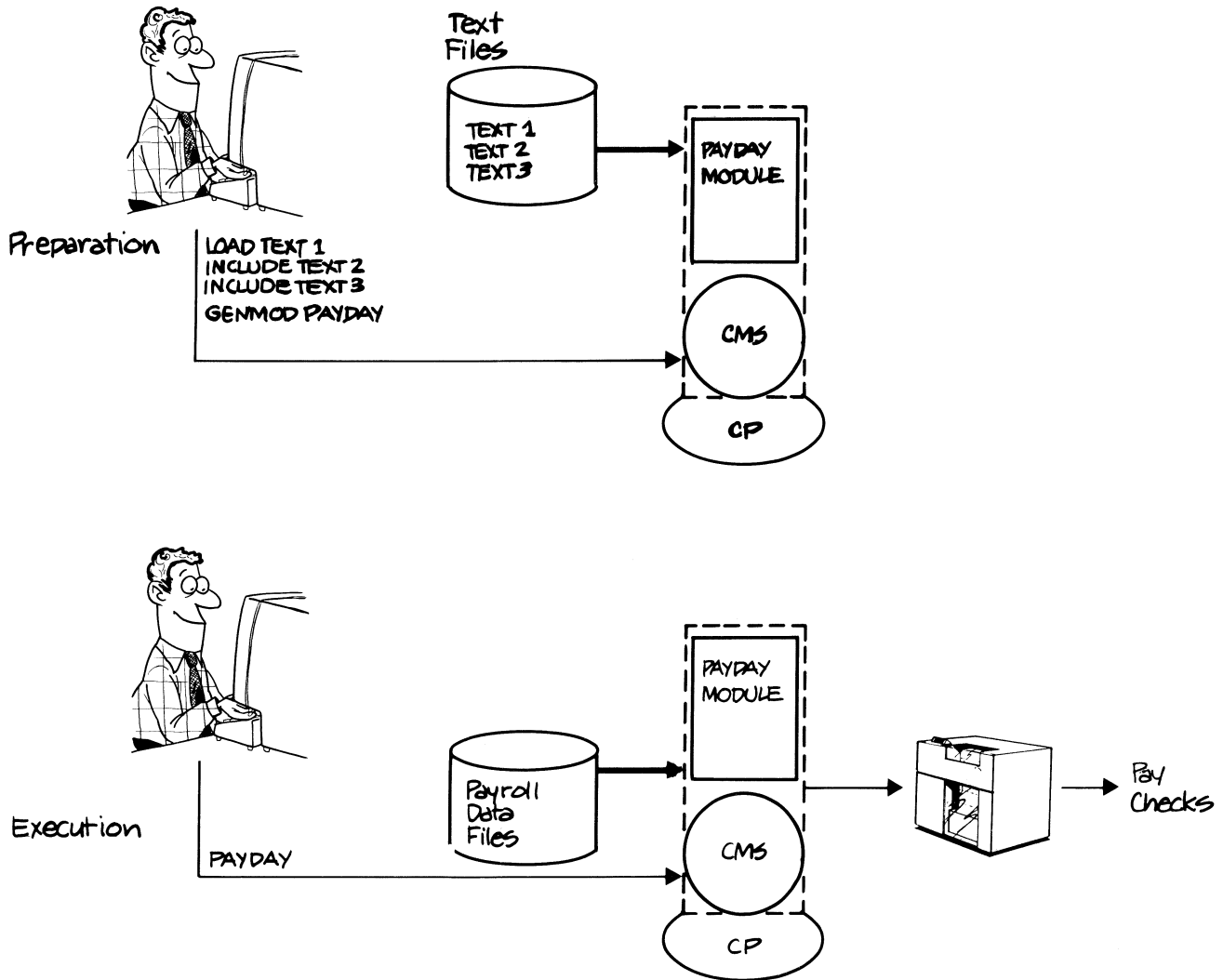


Figure 14. Generating and Executing an Application Program

CMS sometimes needs to be told in which system libraries to look for files used during program execution. For example, TEXT files resulting from a compile process may be available only from system TXTLIB (TEXT file library) members. Before a TXTLIB member can be loaded, it must be identified to CMS (GLOBAL command). Figure 14 illustrates the building of an executable module for running a payroll job, then executing that module to print the checks.

A Command Recap

From the time a source program is ready to be tested to the time the application program is executed, some of the CMS commands used are:

- **GLOBAL** — tells CMS where to find any routines needed when processing source code during program development. Also tells CMS which system library members to search for a compiled program or for other files needed during program execution.
- **FILEDEF** — defines each input or output file referred to by a program.
- **ASSEMBLE**, **PLIOPT**, etc. — invokes the programming language processor to convert the source code into relocatable object code in a **TEXT** file.
- **LOAD/INCLUDE** — each loads a **TEXT** file into a virtual machine. The **LOAD** command is used to load a single **TEXT** file or the first **TEXT** file of a group that are to make up a module. Use the **INCLUDE** command to load the remaining **TEXT** files of a group. The reason for using the **INCLUDE** command is that it does not reset internal loader tables that establish linkage between **TEXT** files.
- **START** — can be used to begin execution of the object code after it has been loaded into virtual storage.
- **GENMOD** — combines one or more loaded **TEXT** files into a single **MODULE** file (filetype of **MODULE**). The combined program can then be executed by issuing the name of the module.

Processing Programs with the RUN Command

The complete process of program compilation and execution can be requested by issuing the **RUN** command under CMS. The procedure that results examines the filetype of the file to be processed to determine what action is to be taken. Barring an error that interrupts processing, the procedure does all steps needed for program execution.

For example:

- If the filetype of the file to be processed is **TEXT**, the procedure simply loads the file and begins execution.
- If the filetype is **COBOL**, **FORTRAN**, etc., the procedure first invokes the proper compiler to produce an object program. The procedure then loads the object program and begins execution.

The **RUN** procedure allows the passing of operands to a program at execution time.

Putting VM/SP to Work

Reference

VM/SP CMS for System Programming, SC24-5286, contains information about executing programs.

VM/SP CMS Command Reference, SC19-6209, describes the LOAD, RUN, and START commands.

Commands that invoke compilers (for example, PLIOPT) are described in reference manuals for the respective compilers. Some of these compilers are listed earlier in this chapter under “Developing Application Programs under CMS” on page 44.

Processing Jobs in Batch Mode under CMS

Overview

- The system operator creates and manages the CMS batch virtual machine.
- The batch virtual machine is shared by all users for program execution.
- The batch virtual machine executes programs one at a time, in the order it receives the programs.
- Along with the program to be executed, a user must include some job control statements in the job stream to the batch virtual machine.
- Unless requested to do otherwise, the batch virtual machine distributes output to the submitting user.
- An IBM program offering, VM Batch Subsystem, runs within VM/SP.

What the CMS Batch Facility Is

The CMS batch facility is a virtual machine available to all users on that system. The system operator creates the CMS batch facility virtual machine using a special batch userid. The operator then puts the virtual machine in batch mode by issuing the CMSBATCH command. The batch facility virtual machine is available to all VM/SP users for executing programs that would otherwise tie up their virtual machines. System users share the facility on a first-in first-out basis. To make sure that each job starts with a clean virtual machine, the batch machine does device housekeeping and reloads CMS between jobs.

Using the CMS Batch Facility

An application program runs in the batch virtual machine much the same as it would in the user's own virtual machine. The program is sent, along with some control statements, to the batch userid (virtual machine). The batch userid is available from the system operator who initiates and manages the batch virtual machine. Either a real system card reader or a virtual punch spooled to the batch userid can be used to send a program to be executed.

The first records in the job stream must be:

Putting VM/SP to Work

- Job control statements (for example, the JOB statement)
- File definition (FILEDEF) statements, if the program refers to input or output files
- LINK and ACCESS statements if the batch virtual machine must refer to input files residing on the user's direct access storage
- SPOOL statement if the printed or punched output is to be routed to a destination other than the submitting user.

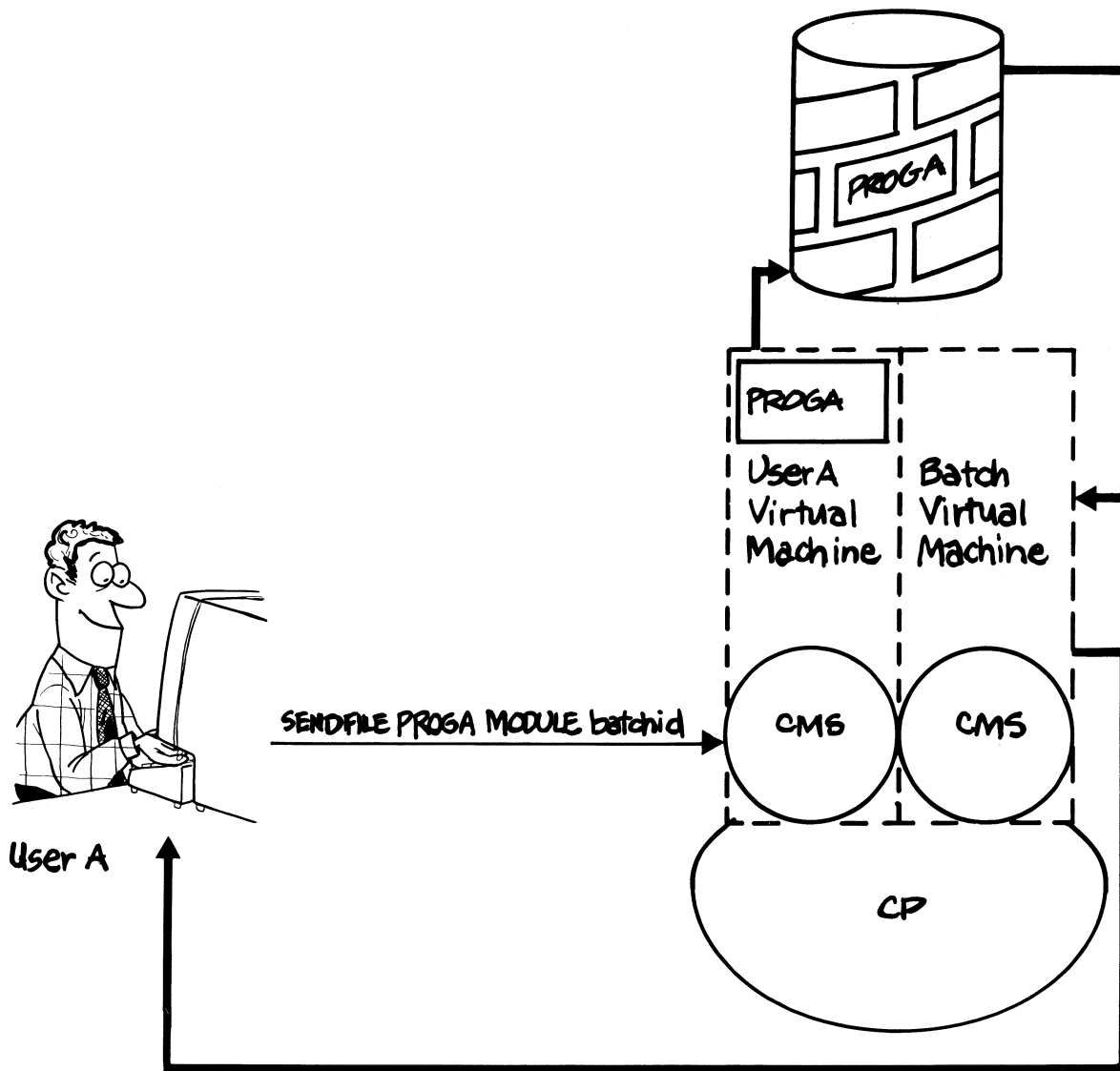


Figure 15. Running Application Programs in the Batch Virtual Machine

Some CP and CMS commands can be included in the batch job stream to control execution in the batch virtual machine. Or the name of an EXEC containing these commands can be passed in the job stream. *VM/SP CMS User's Guide* discusses restrictions on using CP and CMS commands in batch jobs.

The batch facility automatically collects console data in a spool file. Output files resulting from program execution are normally routed as follows:

- Printer files and spooled console output, identified with the submitter's userid, go to the system printer
- Punch files, identified with the submitter's userid, go to the system card punch.

The SPOOL command can be used to change the routing of batch machine output files. For example, if a program's printed output and spooled console output is to be distributed to userid SMITHRL rather than the submitter, the control statements that precede the program should include:

```
CP SPOOL PRT FOR SMITHRL
```

The spool operator can use the PURGE or ORDER commands to control batch jobs that have not begun execution. The user who submitted the job can use the CLOSE, PURGE, CHANGE, or ORDER commands to manage spooled batch virtual machine output files before they are processed.

What the VM Batch Subsystem Is

The VM Batch Subsystem is an IBM program offering that customers can order. It schedules, starts, and monitors jobs within VM/SP. Because the subsystem controls many virtual machines that run batch jobs, many jobs can run at one time. The VM Batch Subsystem schedules each job based on the job class, when it was submitted, start-time range, and priority set by the user or operator. When one of the virtual machines set up to run batch jobs becomes free, the subsystem chooses a job and starts it in the virtual machine. The subsystem then monitors each job. It makes sure the job is not stalled (incomplete, yet no longer using the processor,) and that the job doesn't exceed the maximum time limit or number of output records set by the user.

Using the VM Batch Subsystem

A user's job is normally a CMS EXEC file. It may also include data files, which will be placed on the batch machine's minidisk.

With the VM Batch Subsystem, users can easily run their jobs without using job control language (unless submitted through remote job entry or a card reader.) Jobs submitted through remote job entry or a card reader need to start with a control record that states the job owner and id, job class, password, and options. Simple commands allow users to submit and

Putting VM/SP to Work

cancel a job, review a job being processed or waiting to be processed, and check the status of a job.

When a user requests the status of a job, the subsystem responds with the job options, the current or final status, and a summary of the job's use of system resources.

An installation can extend the VM Batch Subsystem to meet its needs. Some of the tasks that the subsystem allows user-written routines to do are:

- Process user-defined commands
- Audit and change control information (for example, the account number that a user assigns to a job)
- Cause a job or command to be rejected and report the reason to the user who submitted the job or command
- Keep statistics on subsystem use by each user
- Detect whether input through remote job entry is from an acceptable source
- Control which users can issue certain subsystem commands.

Reference

VM/SP CMS User's Guide, SC19-6210, describes the CMS batch facility in detail.

VM/SP CMS for System Programming, SC24-5286, describes the CMS batch facility from the system programmer's point of view.

VM/SP Operator's Guide, SC19-6202, describes the CMS batch facility from the system operator's point of view.

VM Batch Subsystem Program Description/Operations Manual, SH20-2652, tells how to install and use the VM Batch Subsystem.

Running Operating Systems in a Virtual Machine

Overview

- A variety of operating systems can be run in a virtual machine.
- Using an operating system in a virtual machine is much like using that operating system in a real CPU.
- Other than sharing system resources, operating systems running in separate virtual machines run independently of each other.
- The new operating system can be installed and tested in a virtual machine while the old system continues to function.

CP creates a virtual machine for each user who logs onto VM/SP. This virtual machine is of limited use until the user loads an operating system to control work flow. There are many operating systems that run in virtual machines. CMS is the most commonly used because it can be used for so many types of jobs. Other operating systems that run in a virtual machine include VSE/SP (and earlier versions of this DOS-based family of products) and MVS/SP (and earlier versions of this OS-based family of products). Even VM/SP can be run in a VM/SP virtual machine. In Figure 16 on page 66 one user is preparing to run MVS/SP while a second user prepares to run VSE/SP in a virtual machine.

Running an operating system in a virtual machine is a way to allow a VM/SP user to work in that operating system's environment. For example, a VM/SP user might load (IPL) a VS1 operating system into a virtual machine, then operate in the VS1 environment while sharing the VM/SP system with other users.

The migration process under VM/SP may involve:

- Moving to a new level of the base VM/SP system
- Moving to a new level of the operating system that is to be run in a virtual machine
- Replacing the operating system that is to be run in a virtual machine VM with a different type of operating system.

The migration process may include lengthy system testing and debugging. On some non-VM/SP systems this could mean that the system would be unavailable for productive work during installation and testing. Using VM/SP, the new operating system can be installed and tested in a virtual machine while the old system continues to function.

Putting VM/SP to Work

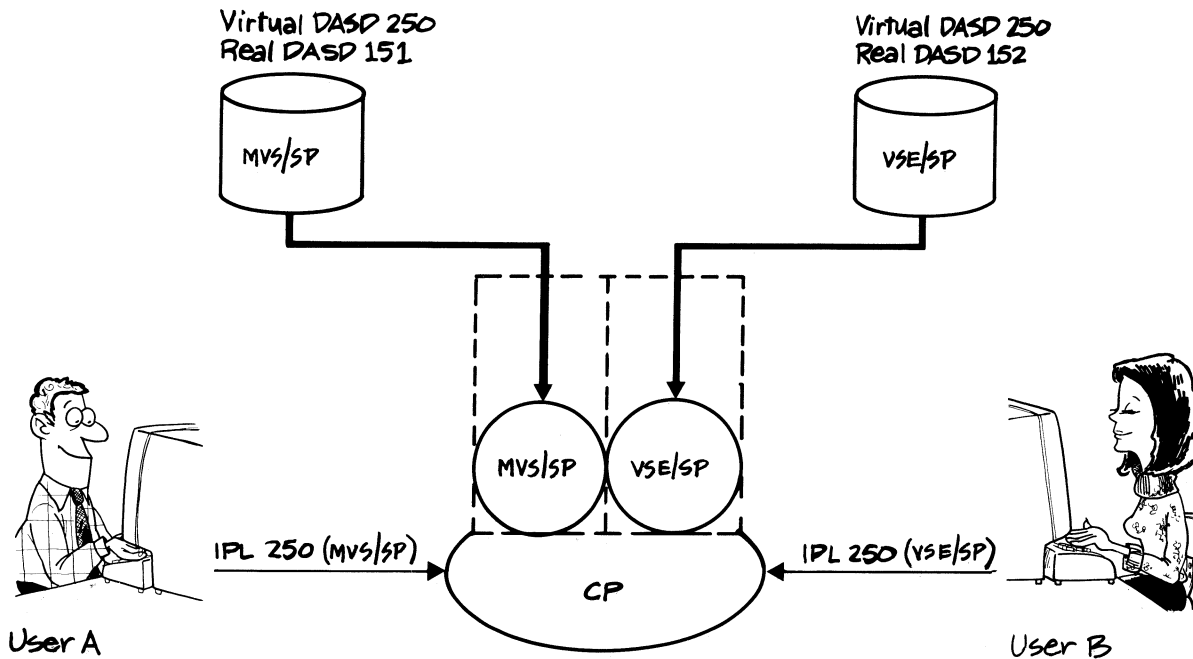


Figure 16. Loading Operating Systems into Virtual Machines

Using an operating system under VM/SP consists of:

- Logging onto VM/SP
- Loading (IPL) an operating system into a virtual machine
- Using the operating system as if it were not running under VM/SP.

The operating system's command set can be used as though the system were running in native mode (not running under VM/SP).

Reference

Running Guest Operating Systems, SC19-6212, describes running operating systems other than CMS in a virtual machine.

VM/SP CMS User's Guide, SC19-6210, describes the use of CMS as an operating system in a virtual machine.

Some Other Programs That Run under VM/SP

Overview

As already pointed out, a variety of operating systems can be run in virtual machines under CP to manage work flow. Other programs can be run in virtual machines to provide functions beyond those of the VM/SP program offering. Examples of these programs are:

- Remote Spooling Communications Subsystem (RSCS) Networking
- SQL/Data System
- VM/Pass-Through
- User-written programs
- Advanced Communications Function for VTAM (ACF/VTAM) for VM/SP.

So far this chapter has described putting a virtual machine to work from the perspective of users doing tasks such as:

- Writing and testing programs
- Executing application programs
- Creating and editing CMS files
- Managing real and virtual system resources
- Using various operating systems in virtual machines.

The list of functions that a virtual machine provides can be extended through use of additional licensed programs or operating systems that run under VM/SP. In addition, users may write their own programs for extending VM/SP's capability.

On the other hand, there is a growing list of programs other than operating systems that can be run in a virtual machine. The primary purpose of these programs is to provide additional functions that extend the capability of VM/SP. Many of these are available from IBM under separate license agreements. This book refers to these programs as licensed programs. "Chapter 5. Extending VM/SP's Capability" later in this book presents an overview of some licensed programs that run under VM/SP. Some examples of these programs are:

- Remote Spooling Communications Subsystem (RSCS) Networking

Putting VM/SP to Work

RSCS Networking transmits messages and data files among different systems in a network.

- SQL/Data System

SQL/Data System manages access to structured user data in a computer.

- VM/Pass-Through

VM/Pass-Through enables users to interactively access other systems in a network of systems.

- User-written programs

Users can write their own programs to provide special functions in VM/SP. These functions can add to those provided by VM/SP or they can replace portions of VM/SP programming.

- Advanced Communications Function for Virtual Telecommunications Access Method (ACF/VTAM) for VM/SP

ACF/VTAM enables users to participate in a Systems Network Architecture (SNA) network. When used with GCS, there is no need to maintain a guest OS/VS1 or VSE operating system.

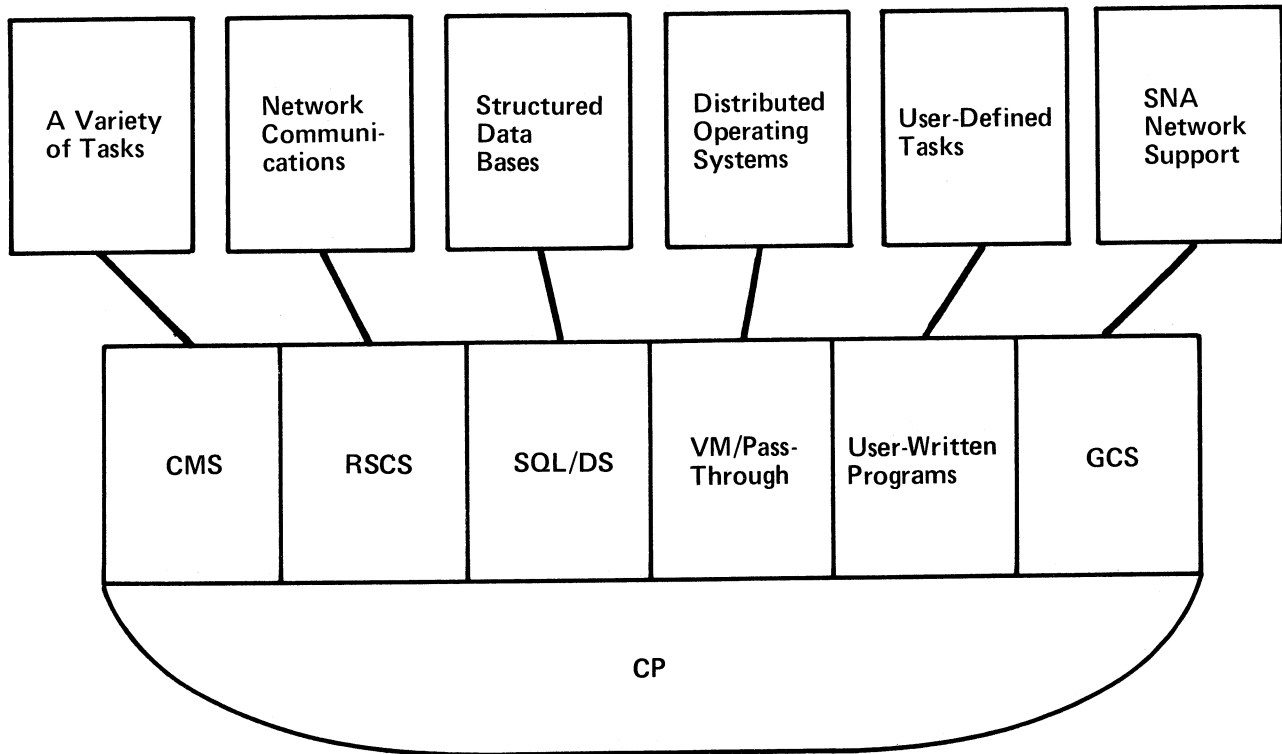


Figure 17. Some Programs That Run in Virtual Machines.

Reference

Chapters later in this book discuss some of the licensed programs that run under VM/SP. Each of these programs has its own documentation.

VM/SP CMS for System Programming, SC24-5286, discusses writing applications to run under VM/SP.

Putting VM/SP to Work

Chapter 3. A Closer Look at CP

This chapter deals with how specific CP facilities affect the VM/SP user. It includes information about:

- What CP facilities are available
- How these facilities can help the user
- How to make these facilities available.

It does not include detailed instructions on how to use CP facilities.

After reading this chapter you will be better prepared to use VM/SP library books such as:

- *VM/SP CP for System Programming*
- *VM/SP Operator's Guide*
- *VM/SP CP Command Reference.*

Chapter Contents

Communicating with CP	73
User Classes Determine System Use Privileges	74
System Directory Entries Help CP Create Virtual Machines	74
Defining and Changing Virtual Machine Configuration and Status	77
System Directory Entries Define Virtual Machines	77
CP Commands Help Manage the Virtual Machine	78
Virtual Console Management	79
CMS Virtual Machines Only	79
Inter-User Communications Vehicle (IUCV)	82
Some Basics about IUCV	82
Communication between Virtual Machines	83
Communication between a Virtual Machine and a CP System Service	84
Hardware Devices That Handle VM/SP Input/Output	85
Defining a System Device Configuration	85
Planning Device Configurations for VM/SP	85
Assuring Data Integrity	87
CP Checks Authorization to Use the System	88
Accessing Another User's Virtual DASD	88
CP Helps Protect Real Storage From Unauthorized Access	88
What If Abnormal Problems Interrupt a Session?	89
Performance Options Can Affect VM/SP Execution	90
Diagnosing System Programming Problems	95

Communicating with CP

Overview

- CP receives information to help it manage real resources and the virtual machine environment from:
 - Entries in the system directory
 - General user commands
 - Commands from users with privilege other than general user, such as operator and system programmer.
- Each CP command falls into one or more privilege classes that define who can issue it.
- An installation has the option of increasing the number of classes from the original 8 to as many as 32, and defining which commands will be in each class.
- A user can issue CP commands from only the privilege class(es) that user is authorized to use.
- Privilege to use certain CP commands is assigned to a user by an entry in the system directory.

To manage system resources and provide user virtual machines, CP must have information made available to it. For example, CP must be told how to allocate resources to users, how to handle virtual machine input and output, and what operating systems users would like loaded (IPLed) into their virtual machines. CP uses data from system directory entries or commands from users to guide it in resource management and system operation.

In general, CP is involved in three levels of system resource management and system operation:

- Creation of the virtual machine environment
- Control of real system resources by the system operator
- Control of individual virtual machines by users.

Certain CP functions are available any time the user is logged onto VM/SP. For example, to request status information about the system or about a virtual machine, the CP QUERY command can be issued from a virtual

A Closer Look at CP

machine running under CMS as well as under CP. However, CP does not recognize CMS commands or commands of other operating systems.

User Classes Determine System Use Privileges

The most common way to communicate to CP is through commands supplied for that purpose. CP commands are grouped according to user privilege classes (Figure 18 on page 75). A user is assigned one or more classes (A-G) in the system directory. The user can enter commands from those assigned privilege group(s) only. For example, general users are assigned privilege class G in the system directory. These users can issue any class G commands. On the other hand, system operators are assigned privilege class A. They can issue any class A commands, which are not available to a general user. Class ANY commands (LOGON, for example) are available to all users. An installation can extend the number of user classes from 8 to as many as 32 by using class override control statements. These statements enable the installation to assign unique classes to specific commands.

System Directory Entries Help CP Create Virtual Machines

Not all virtual machine configurations are alike. A virtual machine is the equivalent of a customized system dedicated to one user. Because each user has different needs, there is a wide variation in the system resources chosen by individual users.

During the VM/SP system generation procedure, control statements that define a virtual machine for each planned user are entered into the system directory.

Each time a user logs onto VM/SP, CP creates a virtual machine based on the user's system directory entries. In addition to common directory entries such as userids, passwords, and default virtual storage size, control statements often define less-common virtual machine characteristics. For example, if a device is to be reserved for exclusive use by one user on a regular basis, a DEDICATE control statement can be included for that user. CP then automatically dedicates that device to that user's virtual machine each time the user logs onto VM/SP. This can be especially helpful for virtual machine guest operating systems running applications that require specific input or output devices.

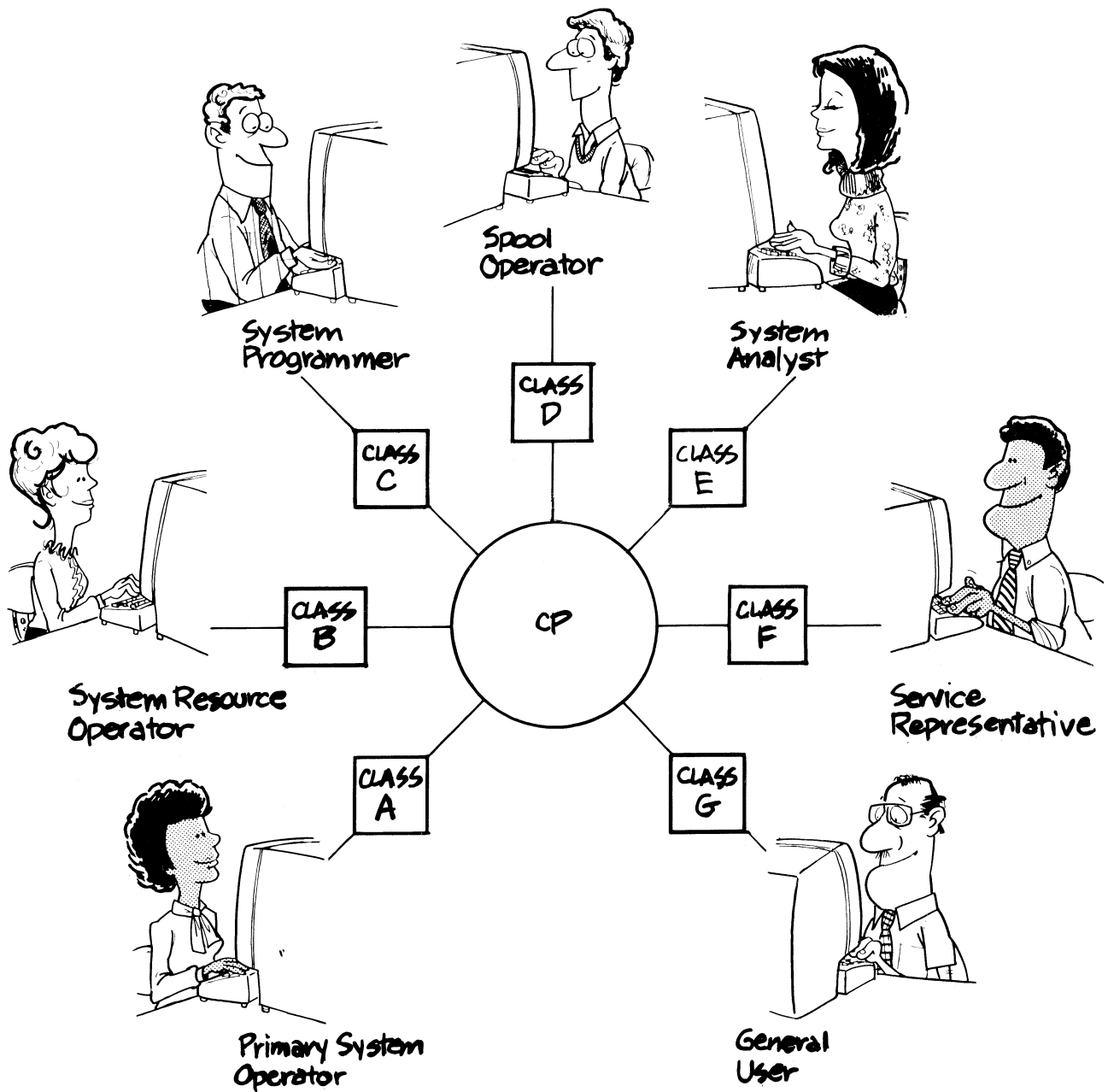


Figure 18. A Command Class for Each Type of User

A Closer Look at CP

Reference

VM/SP Planning Guide and Reference, SC19-6201, describes how to plan and generate control statements for the system directory, and how to plan changes to the user class structure.

VM/SP CP for System Programming, SC24-5285, describes how to make changes to the VM/SP system directory and to the user class structure. It also describes how to add new CP commands and subcommands.

VM/SP CP Command Reference, SC19-6211, describes commands available to the general user for communicating with CP.

VM/SP Operator's Guide, SC19-6202, describes CP commands available to users for communicating with CP.

Defining and Changing Virtual Machine Configuration and Status

Overview

CP determines virtual machine characteristics from:

- System directory entries
- Commands issued by the user to CP during a VM/SP session.

System Directory Entries Define Virtual Machines

The system directory is normally created at VM/SP system generation using the VM/SP directory service program. The directory program takes information from control statements in a user-prepared CMS file and writes it to the system directory. This file can be created using the System Product Editor, a VM/SP facility for creating and editing CMS files. The file must contain a set of control statements for each authorized user. Some examples of the information to be written to the directory via control statements are:

- USER control statement — userid, password, virtual storage size, user class
- CLASS control statement — defines up to 32 classes for each user
- CONSOLE control statement — definition of the virtual machine console
- MDISK control statement — describes the virtual machine's virtual direct access storage (minidisk)
- SPOOL control statement — specifies the virtual machine's virtual unit record devices.

The person replacing or making certain changes to the system directory must have a user privilege class of A, B, or C and have access to the volume on which the directory resides. A directory maintenance facility, described in Chapter 5, makes the updating of the system directory easier and allows general users to make some changes to information associated with their userids.

A Closer Look at CP

CP Commands Help Manage the Virtual Machine

The QUERY command with the VIRTUAL operand can be used to display at a virtual console the status of virtual:

- Storage
- Unit record devices
- Tape and direct access storage devices
- Display devices
- Console
- Channels
- Communications lines.

The QUERY command can be issued from environments other than CP. It produces different results for each environment. For example, the QUERY command under CMS causes a virtual console to display the status of various CMS functions in that virtual machine.

The DEFINE command can be used to make changes to a virtual machine configuration. For example:

```
DEFINE STORAGE 1M
```

establishes the size of a virtual storage at 1 megabyte. Any changes made to a virtual machine via the DEFINE command are temporary (for the life of the session). In some cases, such as redefining virtual storage size, the user must IPL CMS again in the virtual machine after issuing the DEFINE command. Therefore, the DEFINE STORAGE command should not be issued as part of the PROFILE EXEC procedure executed during IPL of CMS.

Some other CP commands that can affect virtual machine status are:

- DETACH — removes virtual devices from the configuration
- DISCONN — disconnects the virtual console from VM/SP but allows execution in the virtual machine to continue
- LINK — makes minidisks in other virtual machines available to a virtual machine
- SET — turns on or off various virtual machine functions, such as performance options, program function (PF) key functions, and message handling

- **TERMINAL** — controls virtual console functions, such as character translation, command line highlighting, maximum line length, and special purpose characters (symbols).

The use of two of these commands is illustrated in Figure 19 on page 80. In one case a user no longer needs the magnetic tape device in the virtual machine configuration. Therefore, it is being freed by using the **DETACH** command. In the other case, a user is issuing the **DEFINE** command to increase the size of that machine's virtual processor storage.

Virtual Console Management

CP's single console image facility allows one user, operating in a single virtual machine, to control concurrently multiple disconnected virtual machines of other users. The CP command language allows the user to easily disconnect a console from one virtual machine and connect it to another in a serial fashion. A virtual machine continues to process programs when its console is disconnected. The user of the controlling virtual machine is called the secondary user. The users of the controlled virtual machines are called the primary user. The secondary user must be identified at system generation on a **CONSOLE** directory control statement for each of the affected primary users. When the primary user disconnects and the secondary user is logged on, the secondary user receives control of the primary user's virtual machine.

CMS Virtual Machines Only

For sessions running under CMS, an **EXEC** procedure can be used to supply information to CP to help define the CMS virtual machine environment. To run automatically during IPL of CMS, the procedure must be named **PROFILE EXEC** and reside on the user's virtual disk at virtual address 191 (A-disk).

The purpose of the procedure is to automatically issue commands that would otherwise be issued manually at the start of each session. These are the commands that do such things as:

- Describe terminal characteristics
- Set up virtual unit record device spooling
- Initialize macro and text libraries
- Access permanent direct access minidisks
- Set program function key operations
- Establish linkage minidisks in other virtual machines.

A Closer Look at CP

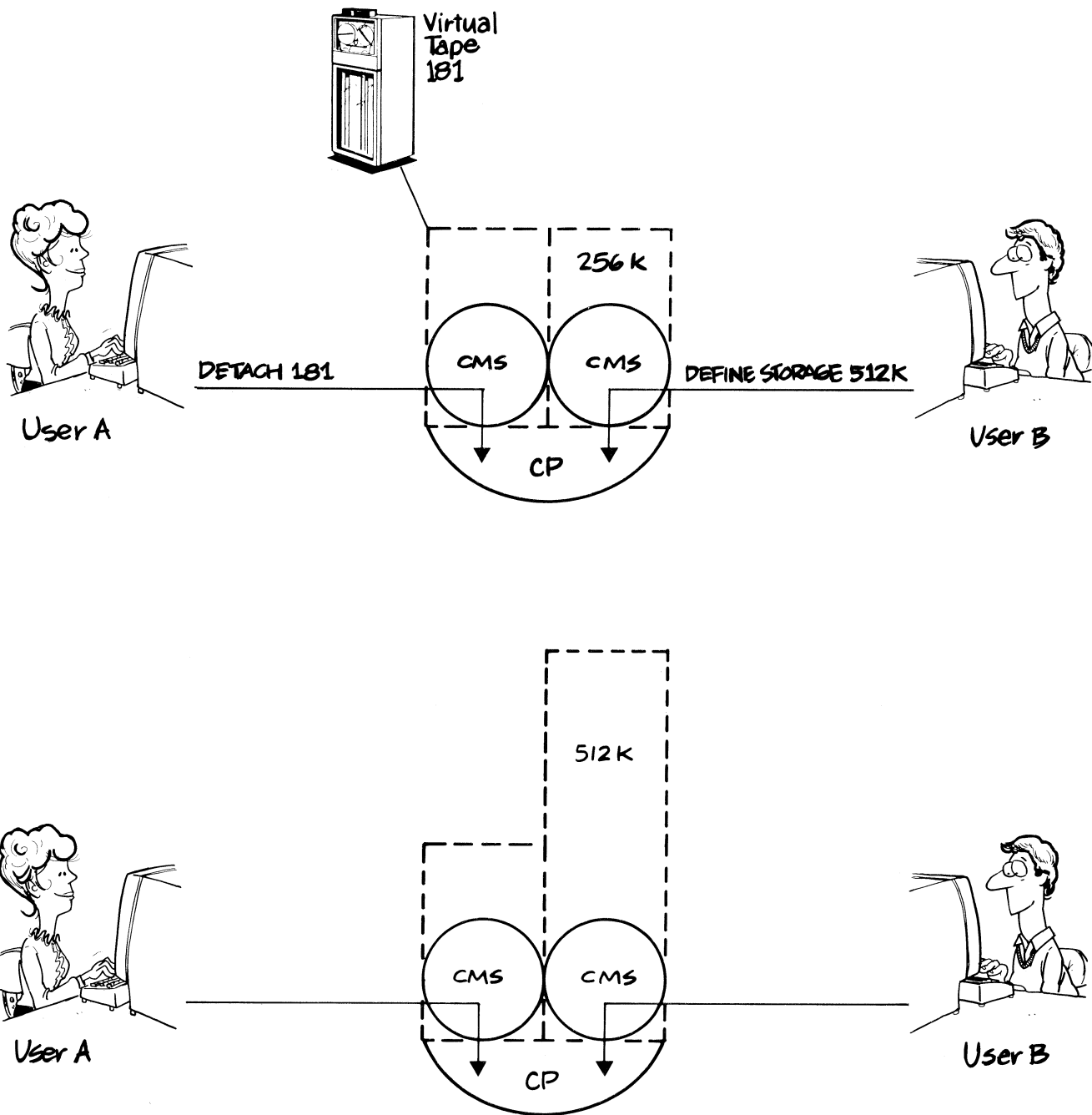


Figure 19. Changing Virtual Machines

Reference

VM/SP Planning Guide and Reference, SC19-6201, describes the directory control statements that define virtual machines.

VM/SP CP Command Reference, SC19-6211, describes commands that a user can issue to change the characteristics of a virtual machine.

VM/SP CMS User's Guide, SC19-6210, discusses the use of PROFILE EXEC during system initialization.

Inter-User Communications Vehicle (IUCV)

Overview

The information in this topic is primarily of interest to programmers.

- **Inter-User Communications Vehicle (IUCV) manages communication among programs in virtual machines on the same VM/SP system.**
- **IUCV also manages communication between a virtual machine and a CP system service.**
- **The targets with which a virtual machine may communicate must be defined by an IUCV entry for that virtual machine in the system directory.**

Note: Advanced Program-to-Program Communication/VM (APPC/VM) function is described under “Some Basics about APPC/VM” on page 132.

Some Basics about IUCV

Inter-User Communications Vehicle (IUCV) enables users to communicate over defined paths within a VM/SP system. Communication is initiated through use of IUCV macro instructions in programs running in virtual machines. To use IUCV facilities, virtual machines must be authorized to establish communication paths. Communication paths are authorized by including IUCV control statements for affected virtual machines when building the system directory at system generation. Figure 20 shows two kinds of communication paths:

- Virtual machine to virtual machine
- Virtual machine to a CP system service.

Communication over IUCV facilities takes place between a source communicator (the virtual machine program initiating the communication) and a target communicator (the virtual machine receiving the communication). The information communicated is called a message and can be of any length. A send/receive protocol is established through programmed IUCV macro functions in the source and target virtual machines.

CMS support facilitates the use of IUCV functions by multiple programs running in the same CMS virtual machine. The CMS macros HNDIUCV and CMSIUCV:

- Identify IUCV programs to CMS
- Begin or terminate communication with another virtual machine or with CP
- Specify exits for IUCV external interrupts.

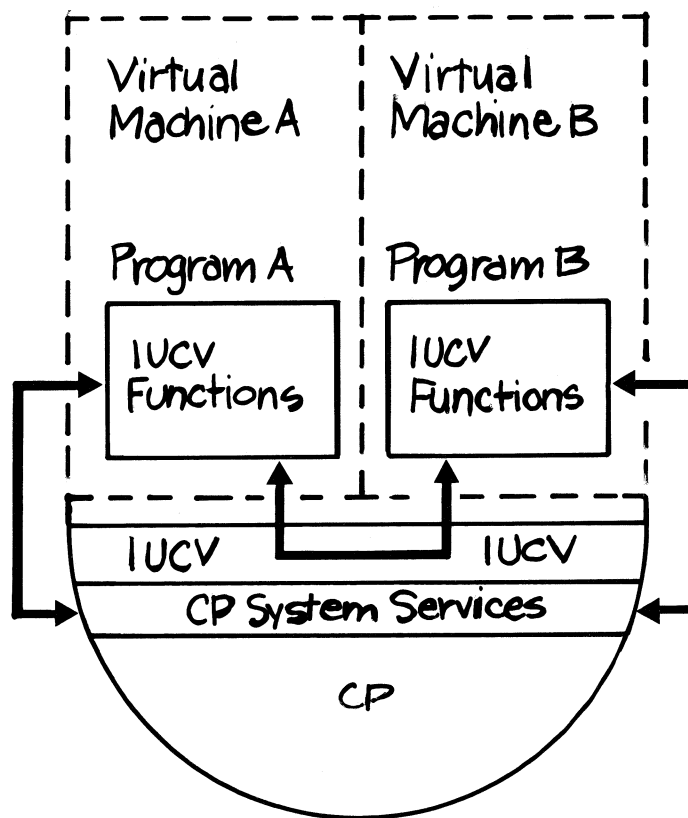


Figure 20. Communication Paths For IUCV Functions

Communication between Virtual Machines

With a communication path authorized in the system directory, two virtual machines can communicate through programmed sequences of IUCV functions. Typical communication tasks performed by IUCV functions include:

1. Each communicator generates an interrupt information buffer (DECLARE BUFFER).
2. The source communicator requests a communication path (CONNECT).

A Closer Look at CP

3. The target communicator completes the path (ACCEPT).
4. The source communicator sends data (SEND).
5. The target communicator receives data (RECEIVE).
6. The source communicator breaks the communication path (SEVER).
7. Drop interrupt information buffers (RETRIEVE BUFFER).

For a more complete description of virtual machine communication via IUCV, see the *VM System Facilities for Programming*.

Communication between a Virtual Machine and a CP System Service

IUCV supports communication between virtual machines and these CP system services:

- Console Communications Services
- Message System Service
- DASD Block I/O System Service
- Signal System Service
- Access Verification System Service
- Message All System Service
- Error Logging System Service
- Advanced Printer Subsystem System Service
- LOGREC System Service
- Collection Resource Management System Service
- Identify System Service.

When a virtual machine program communicates with a CP system service, IUCV routes the communication to the proper CP module to handle it.

Reference

VM System Facilities for Programming, SC24-5288, provides an in-depth description of IUCV.

VM/SP Planning Guide and Reference, SC19-6201, describes the IUCV system directory control statement.

Hardware Devices That Handle VM/SP Input/Output

Overview

- The classes of devices that VM/SP supports are:
 - Direct access storage (DASD)
 - Magnetic tapes
 - Terminals
 - Graphics
 - Unit record
 - Miscellaneous (for example, IBM 3704 and 3705 Transmission Control Units).
- Device support, together with control units and channels, must be defined in a system file (DMKRIO) created at system generation. This includes assigning real device addresses.
- A VM/SP system's device requirements depend to some extent on system facilities and licensed programs that are to be available to the system users.

Defining a System Device Configuration

The VM/SP system generation process includes the preparation of a file that defines that system's real input and output device configuration. That file, DMKRIO ASSEMBLE, consists of groups of device macro instructions.

Planning Device Configurations for VM/SP

Establishing the device configuration for a VM/SP system, although not difficult, does require planning. A minimum configuration is listed in the *VM/SP Planning Guide and Reference*. This serves as a good starting point. Also to be considered are special device requirements for such things as:

- CMS
- Operating systems other than CMS
- RSCS Networking

A Closer Look at CP

- Telecommunications
- Licensed programs running in the VM/SP system
- Special-purpose terminals.

Anyone involved in device planning for a VM/SP system should review related information in the *VM/SP Planning Guide and Reference*.

Reference

VM/SP Planning Guide and Reference, SC19-6201, describes and gives examples of how to define a system device configuration. It also lists all devices supported by VM/SP.

Assuring Data Integrity

Overview

- To log onto VM/SP a user must:
 - be identified (have a userid) in the system directory
 - respond correctly when CP prompts for the logon password.
- Virtual disk space is normally accessible only by the user who owns it. Gaining access to another user's virtual disk space requires a unique (readshare) password.
- Storage keys protect information in real storage from unauthorized use.
- Users can prevent a virtual machine from accessing a shared segment of storage that another virtual machine has modified.

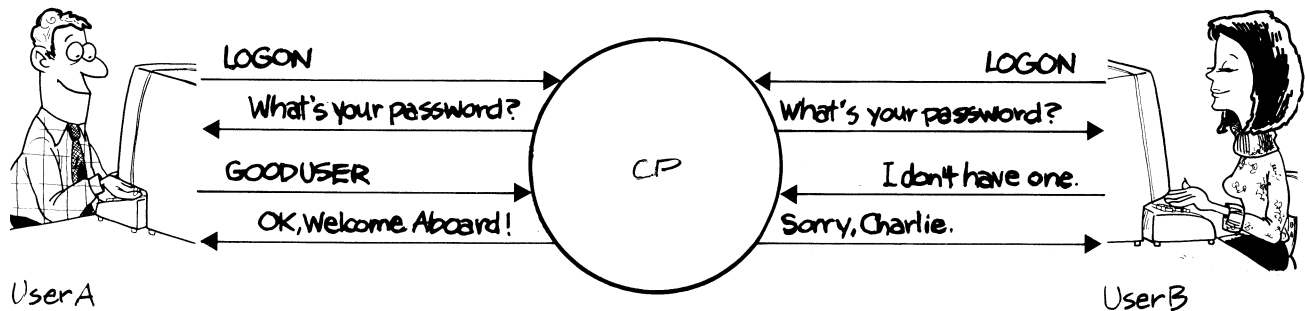


Figure 21. CP Protects against Unauthorized System Use

Because many people can share real system resources, each user's data must be protected from unauthorized access. Data must also be protected against accidental loss caused by system failures. VM/SP provides several functions designed to assure user data integrity. However, there is a limit to how much VM/SP can do in these areas. Each user must be aware that work environment and procedures contribute, either favorably or unfavorably, to the protection of sensitive information.

A Closer Look at CP

CP Checks Authorization to Use the System

To make sure that each person attempting to log onto VM/SP is an authorized user, CP requires that:

1. the logon include a valid userid (one that is included in the system directory)
2. the user respond correctly to a prompt for a password.

At the option of an installation, passwords can be masked to be invisible when typed in on the screen.

An automatic deactivation feature allows the system administrator to prevent the use of certain restricted passwords. Users are not allowed to establish and use these passwords.

Accessing Another User's Virtual DASD

To gain access to files in another virtual machine, a user must get a special password called the "readshare password" from the owner of that virtual machine. That password must be included with the LINK command that establishes a communication path between a virtual machine and a minidisk in a different virtual machine. Permanent linkage to other virtual machines can be established through entries in the VM/SP system directory.

For virtual machines running under CMS, virtual machine linkage can be established through statements in a PROFILE EXEC procedure. Likewise, linkages to other minidisks can be established through statements in a PROFILE GCS procedure for virtual machines running under GCS. There is more information about PROFILE EXEC under "Communicating with CMS" on page 99 later in this book. The *VM/SP Installation Guide* has information about PROFILE GCS.

Finally, users can issue commands necessary to establish virtual machine linkage from terminals during VM/SP sessions.

CP Helps Protect Real Storage From Unauthorized Access

CP helps protect information in real storage from unauthorized program access through use of storage protection keys. Each 2K area of protected storage has a 4-bit key. Programs that access storage also have keys assigned. The program and storage keys must match before the program can access a protected storage area.

Specified segments of real storage can be designated as "shared" at system generation. This means that several virtual machines can share a single copy of these segments of real storage. Sometimes problems can result if any virtual machine is allowed to modify a storage segment being shared by other virtual machines. The default value of a system generation option protects virtual machines from using a modified copy of a shared segment of

real storage (see the NAMESYS macro instruction in *VM/SP Planning Guide and Reference*).

What If Abnormal Problems Interrupt a Session?

Any system is subject to serious problems over which a user has no control. When this happens, the user might be in the middle of a session and not be able to store work files or log off VM/SP before the session terminates. This could cause serious problems. The most obvious is the loss of newly generated data not yet permanently recorded on direct access virtual storage. If a serious system problem occurs, users seldom have an opportunity to log off from a VM/SP session. That could be like leaving the door to a virtual machine open to anyone having access to the system.

Fortunately, CP facilities and user procedures can deal effectively with these problems. CP assures that interrupted sessions are terminated before the system is again available for use. In addition, a system generation option (VMSAVE operand of the OPTION directory control statement) allows an installation to assure that, in the event of abnormal termination, a virtual machine's register and storage contents are saved.

A CMS facility for saving working file data when a System Product Editor session terminates abnormally is described under "VM/SP System Product Editor" on page 110.

Reference

VM/SP CMS User's Guide, SC19-6210, discusses VM/SP features that help protect user data from unauthorized access.

VM/SP CP for System Programming, SC24-5285, describes storage protection keys and shared segment protection.

VM/SP Planning Guide and Reference, SC19-6201, describes system generation macro instructions that affect storage access.

Performance Options Can Affect VM/SP Execution

Overview

The information in this topic is primarily of interest to system programmers.

- **Many factors affect the performance of an operating system in a virtual machine.**
- **VM/SP has several options that, under certain conditions, improve system performance.**

A detailed discussion of performance factors is beyond the scope of this book. The concepts and terminology associated with performance improvement are not intended for the general user and may require additional reading for any user.

Among the factors that affect system performance are:

- The processor model
- The total number of virtual machines in use
- The type of work being done in each virtual machine
- The paging device
- The availability of real storage
- The VM/SP performance options in effect.

VM/SP performance options include:

- Favored execution
- User priority
- Reserved page frames
- Virtual = real
- Affinity
- Multiple shadow table support
- Single processor mode

- Queue drop elimination
- Virtual machine assist
- Extended control-program support.

Most performance options can be specified at VM/SP system generation or through operands of the SET command under CP. Two CP commands for determining system performance are:

MONITOR - (privileged) controls the recording of events that occur in the real VM/SP system.

INDICATE - (general use) displays at a virtual console the use of, and contention for, major system resources.

Automatic collection of data related to system performance can be requested with the SYSMON macro instruction in the DMKSYS file at system generation.

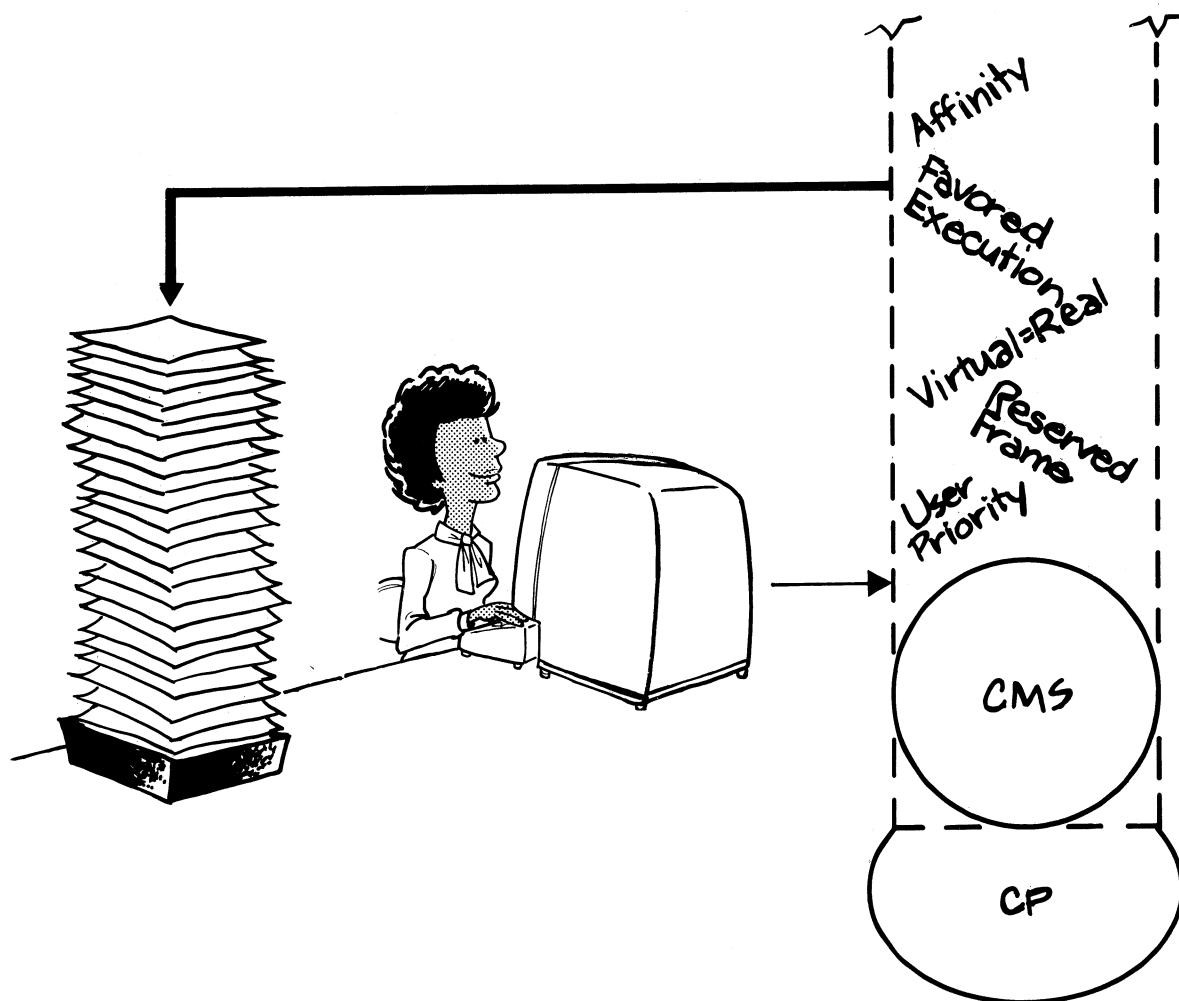


Figure 22. Performance Options Can Improve Productivity.

A Closer Look at CP

Favored Execution

The favored execution option allocates more system resources to a given virtual machine than it would have under normal operation. The system operator specifies this option through operands of the SET command under CP.

User Priority

The system operator uses the PRIORITY operand of the SET command to assign relative dispatching priorities of virtual machines on a VM/SP system. A virtual machine with a high priority (lower number) gets a larger share of system resources than one of lower priority (higher number).

Reserved Page Frames

The system operator uses the RESERVED operand of the SET command to reserve page frames for exclusive use by a given virtual machine. This can improve the performance of the affected virtual machine by reducing that machine's contention for page frames.

Virtual = Real

The virtual = real option can be specified at VM/SP system generation. It is specified on the OPTION directory control statement for any virtual machine whose virtual storage pages (other than page zero) are to be locked into fixed positions of real storage. The virtual addresses and the real addresses of these storage positions are equal. This eliminates CCW translation (except for page 0) and paging for the affected virtual machine. As part of its system management functions, CP relocates page 0.

Affinity

The affinity option enables virtual machines that run on attached processor (AP) or multiprocessor (MP) systems to select a processor for program execution. This option can be selected at system generation time using the AFFINITY operand of the OPTION control statement. The system operator (or user authorized by an affinity directory entry) activates the option using the AFFINITY operand of the SET command.

Multiple Shadow Table Support

To reduce system overhead when a virtual machine changes control register 1 (CR1) values, VM/SP maintains a queue of segment table origins and associated shadow tables for virtual machines. Thus, each time an MVS or SVS system dispatches a new address space (changes CR1), VM/SP can dispatch the proper shadow table. General users can control multiple

shadow table support through use of the STMULTI operand on the SET command under CP.

Single Processor Mode

Running in single processor mode improves throughput for an MVS (AP or MP) system running under VM/SP. The single processor mode option allows an installation to dedicate a processor to an MVS Virtual=Real (V=R) virtual machine. The system operator turns single processor mode environment on or off using the SPMODE command. *Running Guest Operating Systems* has more information about this option.

Queue Drop Elimination

VM/SP attempts to optimize throughput by dropping idle virtual machines from the active queue. However, in some case virtual machines become active again sooner than expected and must be put back on the active queue. Removing virtual machines from, and restoring them to, the active queue can cause considerable system overhead. The system operator uses the QDROP operand of the SET command to eliminate scanning of a virtual machine's page and segment activity for the purpose of removing inactive pages.

Virtual Machine Assist

Virtual machine assist is a feature on most supported VM/SP processors. It handles SVC interrupts, invalid page conditions, and several privileged instructions. Although virtual machine assist is designed to improve VM/SP performance, it may also result in improved virtual machine performance because it makes more resources available to virtual machine users. The system operator uses the SASSIST operand on the SET command to activate or deactivate the facility.

Extended Control-Program Support

Extended control-program support (ECPS) extends, for certain privileged instructions, the hardware support provided by virtual machine assist. The system operator uses the SASSIST and CPASSIST operands of the SET command to activate or deactivate ECPS.

Reference

VM/SP CP for System Programming, SC24-5285, provides an in-depth discussion of VM/SP performance options.

VM/SP Planning Guide and Reference, SC19-6201, describes system generation of performance options.

A Closer Look at CP

VM Running Guest Operating Systems, SC19-6212, describes the single-processor mode option and virtual machine operator procedures for dynamic SCP transition.

Diagnosing System Programming Problems

Overview

The information in this topic is of interest primarily to system programmers.

- System program problems can usually be classified as one of the four general types.
- Problem debugging is most effective when a systematic step-by-step approach is used.
- VM/SP offers several tools to help diagnose system programming problems.

VM/SP system program problems are generally of four types:

- Program loops
- Wait state
- Abnormal program termination (ABEND)
- Incorrect results.

Problem debugging consists of four basic steps:

- Recognizing that the problem exists
- Identifying the type of problem and the area of the system affected
- Collecting and analyzing data, then sorting out the data that applies to the problem
- Finding and fixing the problem.

VM/SP commands enable the system programmer to:

- Dump storage data
- Display data
- Store data.

The process of diagnosing and fixing system programming problems is documented in the *VM Diagnosis Guide*. Anyone whose job responsibilities

A Closer Look at CP

are to include system programming debugging should become familiar with the related information in that book.

Reference

VM Diagnosis Guide, LY24-5241, describes techniques, procedures, and VM tools for locating and fixing system programming problems.

Chapter 4. A Closer Look at CMS

As pointed out earlier in this book, any one of many operating systems can be used to control work flow in a virtual machine under VM/SP. CMS is the most versatile and most widely used operating system. It runs only in a virtual machine under CP.

This chapter discusses some of the more commonly used CMS facilities. The information in this chapter is intended to help prepare you to use some of the other books of the VM/SP library, such as:

- *VM/SP CMS User's Guide*
- *VM/SP CMS Command Reference*
- *VM/SP System Product Interpreter User's Guide*
- *VM/SP System Product Editor User's Guide*
- *Programmer's Guide to the Server-Requester Programming Interface for VM/SP.*

This chapter is not intended to be used as a reference for *using* CMS.

Chapter Contents

Communicating with CMS	99
Initializing CMS	99
Work Flow Management through Commands and Messages	101
Subcommand Environments under CMS	102
Procedures for Automated Command Execution	103
CMS from the User's Point of View	104
How the General User Sees CMS	104
How Other Users See CMS	105
Command Language Processors under CMS	107
EXEC Procedures Are Sequences of Commands	107
EXEC Procedure Language and Processor in CMS	109
VM/SP System Product Editor	110
Working in the Editor Environment (XEDIT)	111
Preventing Data Loss While Using the System Product Editor ..	112
CMS File System	113
How CMS Files Are Structured	113
How CMS Files Are Identified	115
How CMS Files Can Be Shared Among Users	116
CMS Session Services	118
Virtual Screens	118
Macro Support	119
The System Product Editor	120
CMS HELP Facility	121
Requesting Help	123
The Console Stack Helps Keep Work Flowing	124
Exchanging Data through the Program Stack	124
Terminal Input Buffer Is Temporary Storage for Keyed Data	126
IBM Enhanced Connectivity Facilities on VM/SP	127
Starting Connectivity Facilities	127
Writing Servers	127

Communicating with CMS

Overview

The purpose of communication between the user and CMS is to manage work flow in the user's virtual machine. To handle this task CMS supports:

- A set of user commands
- Subcommands and macro instructions issued from secondary environments under CMS
- Messages that prompt, reply, convey information, or indicate an error condition
- A set of macro instructions and functions for programs written in Assembler language
- Command language processor control statements
- Commands that run only in the CMS/DOS environment
- Programs that compile high-level language programs
- CMS messages in the user's National language (for those languages supported by VM/SP).

Initializing CMS

The first step of a user session is normally the loading (IPL) of CMS into the user's virtual machine. During the IPL CMS process, commands that help define the CMS virtual machine environment may be passed automatically to the system. These commands are usually included as statements in an EXEC procedure named PROFILE EXEC, which resides on the user's primary virtual direct access storage (A-disk). CMS executes PROFILE EXEC, if it exists, as part of the IPL process. PROFILE EXEC might contain commands that do such things as:

- Describe the virtual console
- Define spooling for virtual unit record devices
- Initialize macro and text libraries
- Access virtual direct access devices as part of the virtual machine configuration

A Closer Look at CMS

- Establish linkages to other virtual machines
- Set up program function key operations.

CMS also contains a System Profile feature. The System Profile feature provides an easy method for installations to define the users' CMS environment. It allows system administrators to tailor environments for users at initialization time. It is a flexible tool that can be used to accomplish anything that can be done from within an EXEC.

The environment can be tailored by modifying an EXEC named SYSPROF. A default SYSPROF EXEC is provided in VM/SP. It:

- displays the CMS system id
- issues the initial console read
- processes parameters passed to it via the IPL command
- accesses the 191 and 192 disks
- issues initialization-related messages
- executes the user's PROFILE EXEC (if one exists).

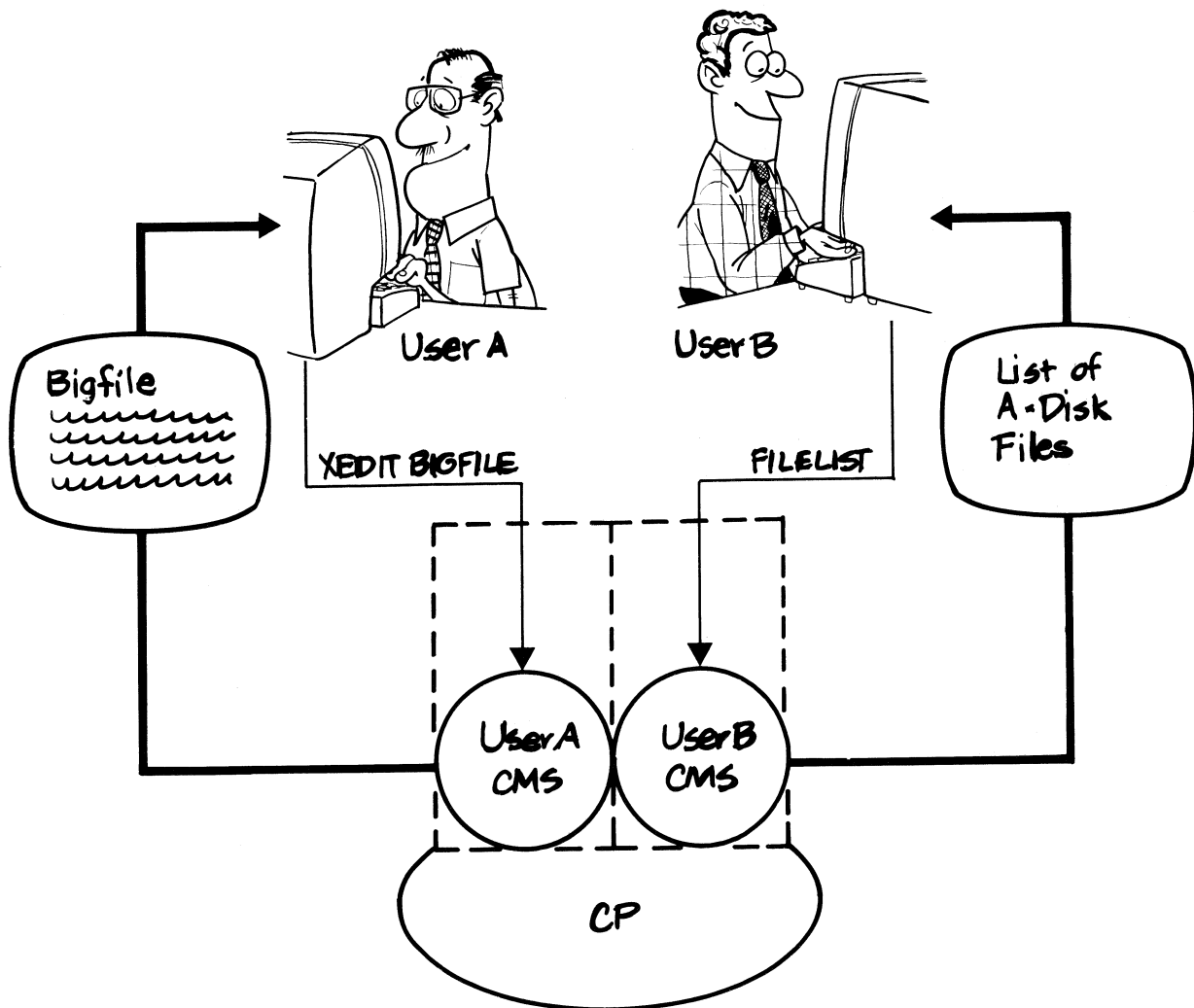


Figure 23. User/CMS Communication

Work Flow Management through Commands and Messages

The term “CMS virtual machine” in this book means a virtual machine in which CMS is the operating system. Chapter 2 discussed several ways of using a CMS virtual machine. No matter how a CMS virtual machine is used, actions are initiated by issuing commands to CMS. For example, a user can:

- Use the System Product Editor (XEDIT command) to create a CMS file named INCOME DIVIDEND by issuing:

```
XEDIT INCOME DIVIDEND
```

A Closer Look at CMS

- Have CMS display the primary virtual direct access storage (A-disk) status by issuing:

```
QUERY DISK A
```

- Read a file from the virtual reader to virtual direct access storage by issuing:

```
RECEIVE
```

To get a good idea of all the things that may be requested of CMS, you should look through the *VM/SP CMS Command Reference*. Figure 23 on page 101 illustrates the use of two CMS commands. One user is initiating an editing session to make changes to a file named BIGFILE. The other user is displaying a list of the files on his virtual A-disk. The second user may then use CMS commands to initiate a variety of actions related to using and managing the displayed files.

Issuing commands to CMS is much like telling another person in English what is to be done. If a user makes a mistake or CMS needs more information, CMS displays appropriate messages. CMS messages convey several types of information, such as pointing out user errors, completion of events, and response to user queries. CMS messages may also prompt for input information needed to complete an action.

If a user is not sure how to request an action, issuing:

```
HELP
```

requests the CMS HELP facility to lend a hand. Another topic later in this chapter describes the CMS HELP facility.

Subcommand Environments under CMS

Special-purpose facilities that support their own sets of commands can be invoked under CMS. The System Product Editor and the CMS Debug facility are two examples of these subcommand environments. CMS does not recognize subcommands of these environments until the user issues the CMS command to cause the virtual machine to enter that environment. However, once a virtual machine is in a subcommand environment, the user can still issue regular CMS, or even CP, commands. For example, before a virtual machine enters the System Product Editor environment, the UP command (an XEDIT subcommand) has no meaning to CMS. However, in the System Product Editor environment, CMS continues to respond to the QUERY READER (CMS) command as well as the UP (XEDIT) subcommand.

Most CMS commands are documented in the *VM/SP CMS Command Reference*. CMS subcommands are documented, by facility, in separate sections of that book, or in separate documentation describing the facility. Some special cases, such as the command to invoke the programmable operator facility (PROP), are documented in the *VM System Facilities for Programming*.

Procedures for Automated Command Execution

Topics earlier in this book mention how a CMS virtual machine environment can be governed by making a PROFILE EXEC available to CMS during the IPL process. During a session other sequences of commands in EXEC procedures can be communicated to CMS by simply issuing the filename of the procedure. For example, an EXEC procedure named SHIPFILE EXEC can be executed by simply issuing the filename:

```
SHIPFILE
```

CMS includes the System Product Interpreter command language processor for executing automated procedures. A topic later in this chapter describes the System Product Interpreter and gives references to related information sources.

Reference

VM/SP CMS User's Guide, SC19-6210, describes communication between users and CMS.

VM/SP CMS Command Reference, SC19-6209, describes CMS commands and subcommands.

VM/SP CMS Macros and Functions Reference, SC24-5284, describes macro instruction and functions.

VM/SP System Product Interpreter Reference, SC24-5239, describes the command language processor for CMS.

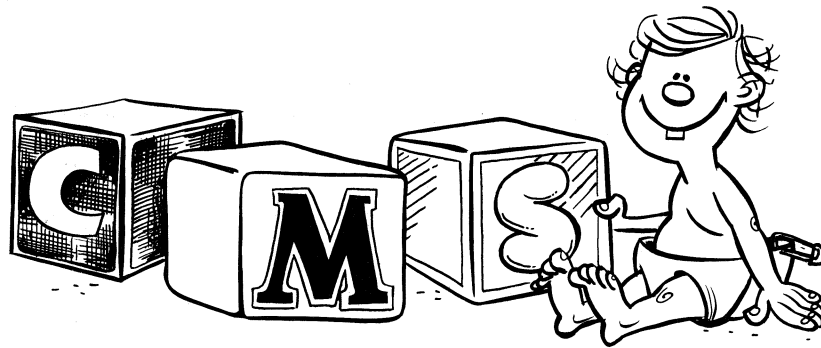
A Closer Look at CMS

CMS from the User's Point of View

Overview

CMS is a general purpose operating system used to do a wide variety of jobs. From a broad perspective these jobs are:

- Plan, install, and maintain the VM/SP system
- Write, debug, and execute programs
- Create and edit data files
- Communicate with other system users.



CMS can help different users in different ways. Its facilities are so diverse that most users never need to use all of them. However, the more any user is aware of CMS's capabilities, the more potentially productive that user becomes.

How the General User Sees CMS

A general user of VM/SP is a person to whom CMS is simply a tool for getting a job done. Most general users are concerned about the end results, not how the system produces those results. This person's view of the system normally includes only a terminal, the data displayed at that terminal, and output such as listings that are printed and delivered as expected.

CMS is quite accommodating to the general user with limited experience. People who know little or nothing about data processing or computers can quickly learn to use CMS. Most user errors in communicating with CMS cause no serious problem. Displayed messages help the user get back on the right track. When not sure about the format of a command, a user can request that CMS display a HELP panel to explain that command. Users can communicate with CMS in their National language if VM/SP is installed with that support.

To the general user CMS means being able to do many tedious and time-consuming tasks more easily and efficiently. These tasks include such things as:

- Writing letters and reports
- Writing and debugging application programs
- Executing application programs
- Communicating with other system users.

How Other Users See CMS

Specialized VM/SP users include the:

- System analyst
- System programmer
- Operator
- Administrator.

These people are responsible for providing a system that is as useful and efficient as possible. CMS can be an important part of meeting those responsibilities.

People responsible for productivity within their organization see CMS as a tool for increasing efficiency. For example, programmers can write, test, and execute programs far more efficiently using CMS facilities than using traditional hands-on methods. Programs can be developed online, making punched card source decks unnecessary. Rather than waiting for batch processing results, programmers can test programs in their virtual machines. They can correct errors interactively from a terminal and immediately retest the program. From the programmer's point of view, this makes programming more enjoyable because it means quick turnaround, better debugging facilities, and less frustration.

The system operator can use CMS commands while performing system management tasks. For example, the operator may need to make changes to VM/SP system files. For this, the operator uses the System Product Editor the same way a general user might use it for making changes to an application program. The system operator also uses CMS facilities such as those invoked by the CMSBATCH, FORMAT, LISTDS, and PRINT commands.

A Closer Look at CMS

Reference

VM/SP CMS User's Guide, SC19-6210, is the most complete information source for users to learn about CMS.

VM/SP CMS Primer, SC24-5236, can help a new user get started using CMS.

Command Language Processors under CMS

Overview

- EXEC procedures can greatly simplify communication between users and VM/SP.
- EXEC procedures range from a single statement to complex programmed sequences of statements.
- CP or CMS commands can be executed from EXEC procedures.
- System Product Interpreter is the VM/SP command language (EXEC) processor for procedures written in the Restructured Extended Executor (REXX) language.

New users sometimes hesitate to get involved with EXEC procedures. Maybe its the name “EXEC” that frightens them. Perhaps they have seen some of the complex procedures developed by experts. In any case, there is really nothing to be frightened of. EXEC facilities are designed to be easy to use and to simplify computer use. EXEC procedures can save a great deal of time, especially for sequences of commands and other statements issued on a regular basis.

EXEC Procedures Are Sequences of Commands

An EXEC procedure is a sequence of commands and other statements that can be processed by CMS. Besides CP or CMS commands, EXEC procedures can contain statements to provide functions such as the testing of variables and conditional branching. The System Product Editor is used for creating EXEC procedures. Although certain format and syntax rules apply, the process is the same as for creating other types of CMS files.

Depending on the complexity of the procedure, various control statements that affect execution can be included. For example, they:

- Execute a portion of the procedure only if certain conditions exist
- Issue informational messages or messages that request input
- Substitute variables supplied by the user into statements in the EXEC procedure.

An EXEC can be of any length, from a single command to a complex set of routines. An EXEC file can have any filename, but some filenames (PROFILE, for example) are reserved for a special purpose. Normally, the filetype of the procedure must be EXEC.

A Closer Look at CMS

Once an EXEC procedure is stored on virtual direct access storage, the procedure can be executed by simply issuing the name of the EXEC file, in the same manner as CP or CMS commands. Thus, each EXEC procedure becomes, in effect, a new command. For example, a procedure whose filename and filetype identifiers are SHIP EXEC can be executed by simply issuing:

SHIP

If the name of an EXEC file is already the name of an existing CP or CMS command, the system normally executes the procedure instead of the system command procedure having the same name. This can be controlled with the IMPEX option of the SET command.

Figure 24 shows a user communicating a sequence of commands to CP by simply issuing LIST, the name of the procedure containing that sequence.

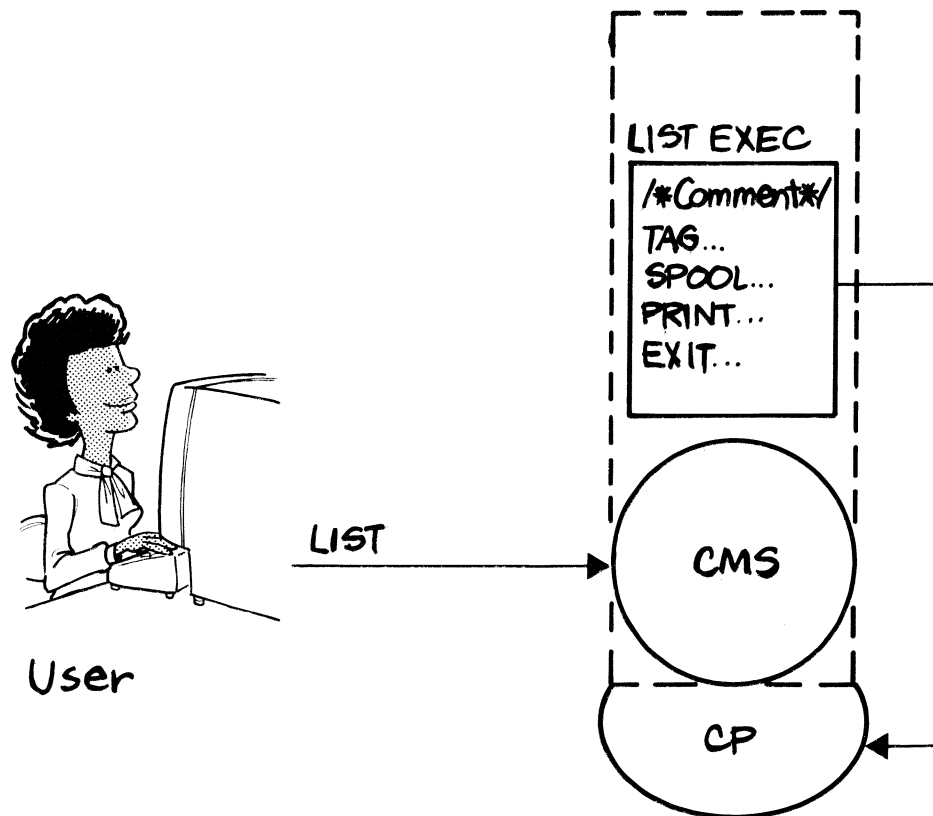


Figure 24. EXEC Procedures Simplify User-to-System Communication.

EXEC Procedure Language and Processor in CMS

Just as any program is written in a language supported by a processor for that language, EXEC procedures for CMS are written in the Restructured Extended Executor (REXX) language. REXX statements are processed by the System Product Interpreter command language processor in CMS.

REXX is designed to be used by anyone, from a novice general user to an experienced programmer. It is a high-level programming language. For example, REXX statements do such things as:

- Execute conditional loops
- Assign values to variables
- If ... then ... else conditional execution
- Do arithmetic operations
- Character conversion and translation
- Compare values.

For the most part REXX statements are written in simple English words and phrases. For example, to display a request for a user to enter the current date, the procedure statement is:

```
say "Please enter today's date."
```

Or, to assign the value of 40 to a variable named "rate," the procedure statement is:

```
rate = 40
```

Reference

VM/SP CMS Primer, SC24-5236, provides information to help users learn to write EXEC procedures.

VM/SP CMS User's Guide, SC19-6210, gives detailed information about creating EXEC procedures for using with CMS. System Product Interpreter, EXEC, and EXEC2 are discussed and differences are explained.

VM/SP System Product Interpreter User's Guide, SC24-5238, discusses and gives examples for using the REXX language.

A Closer Look at CMS

VM/SP System Product Editor

Overview

- The System Product Editor can be used to do a variety of CMS file creation and updating tasks.
- System macros and user-written procedures can be executed from the System Product Editor environment.
- The AUTOSAVE facility protects data against accidental loss caused by abnormal termination of an editor session.



This topic refers to the System Product Editor simply as “the editor.” The editor is a full-screen editing facility that runs under CMS. “Full-screen” means that it allows working with a full video screen of file data displayed at a terminal. This is a considerable advantage over editing programs designed to work with one data line at a time.

Traditionally, the only functions expected of a system editor were those of file creation and updating. The VM/SP editor has many facilities to make these tasks easy and efficient. For example, it has a power input (POWERINP subcommand) facility so the user can key in text as though it were one continuous line. The editor takes care of shifting to new input lines and reformatting words that get split between the end of one line and the start of the next line. It is like typing without ever having to do a carriage return.

A summary of the tasks that the editor can help users do includes:

- Create CMS files
- Make changes to existing files, including simultaneous editing of different files or multiple sections of the same file

- Combine existing files or portions of files
- Place file lines in the console stack
- Sort data in all or part of a file
- Write and use macro procedures
- Request help in using the editor
- Search files for specific data
- Redefine the editor environment.

Working in the Editor Environment (XEDIT)

The user can enter the editor environment by issuing the XEDIT command under CMS. While working in the editor environment a user may create new CMS files or make changes to existing CMS files. For example, to create a CMS file named BICYCLES TOURING (or make changes to an existing file of the same name), the user begins by issuing:

```
XEDIT BICYCLES TOURING
```

The editor then displays a working copy of the requested file on the user's terminal screen. From that point the display screen is like an electronic notebook for making additions or changes.

The System Product Editor (XEDIT) takes advantage of CMS Session Services. (See "CMS Session Services" on page 118 for a description). Users have the option of specifying what window XEDIT should use to display a file. The default window is "XEDIT."

Editor (XEDIT) subcommands provide a variety of ways to manipulate a file on a display screen while making changes. The SET subcommand provides many options for managing the editor environment. Users can execute, from the editor environment, system macro routines invoked by XEDIT subcommands. For example, all or part of a file can be sorted (SORT subcommand), or file lines placed on the console stack (STACK subcommand). Procedures, written in the Restructured Extended Executor (REXX) language, can be invoked from the editor environment by issuing the procedure name.

The user can initiate many editor functions simply by pressing program function keys. Program function (PF) keys are automatically set to perform useful functions in the editor environment. Users can override these settings with the SET PF subcommand, either in a file called PROFILE XEDIT, or by issuing it manually during an editor session. For example, PF10 can be set to display the current tab column numbers by issuing:

```
SET PF10 QUERY TABS
```

A Closer Look at CMS

Thereafter (until the end of that XEDIT session) pressing PF10 causes XEDIT to display the tab column numbers defined by the SET TABS subcommand. The editor session is then resumed by pressing the CLEAR key.

For descriptions and examples of the kinds of things that can be done during an editing session, refer to the books listed at the end of this topic. Normally an editing session is terminated by issuing the FILE subcommand. This causes the working file copy to become a permanent copy on the user's A-disk.

Preventing Data Loss While Using the System Product Editor

Accidental loss of large amounts of work done during an editing session can be prevented by periodically issuing the SAVE subcommand. This causes the current working copy of a file to replace the user's disk storage (normally A-disk) copy of that file. Other than that, the editing session is not affected by the SAVE subcommand.

Users can request the editor to automatically save a copy of the current working file at specified intervals during an editing session. This is done by using the AUTOSAVE option of the SET subcommand. For example, issuing:

```
SET AUTOSAVE 10
```

causes the editor to create a copy of the working file on the user's disk storage and update it each time a combined total of 10 lines are added, changed, or deleted in the working file. The user's permanent copy of that file is not affected by the AUTOSAVE option. In the example just given, the most the user might lose are the last ten changes made to the file.

The editor can help users do many things besides simple file creation and editing. Most VM/SP users must at least occasionally use the editor. This book does not discuss all the editing facilities because the editor has separate and complete documentation in the VM/SP library.

Reference

VM/SP CMS Primer, SC24-5236, instructs a new user on how to use the editor.

VM/SP System Product Editor User's Guide, SC24-5220, describes editor facilities and gives examples for their use.

VM/SP System Product Editor Command and Macro Reference, SC24-5221, describes the format and options for each editor subcommand.

VM/SP CMS User's Guide, SC19-6210, describes CMS Session Services.

CMS File System

Overview

- A file is the working unit of information in the CMS environment.
- A file is made up of records of specified length and format.
- CMS stores files on virtual disk space that has been formatted specifically for CMS files.
- CMS files can be written or read by CMS only.
- CMS files are identified by filename, filetype, and filemode.
- An IBM program offering, VM File Storage Facility, allows users to share CMS files.

How CMS Files Are Structured

Most users' view of the CMS file system is limited to the display of CMS files at their terminals. Because CMS handles file management, usually based on default format options, users normally have little need to be concerned about how CMS stores files on their virtual direct access storage. However, many CMS users must eventually do tasks that require some knowledge of the CMS file system.

Generally speaking, CMS stores each displayed data line as one record in the user's direct access virtual storage. CMS does some tailoring of these lines, depending on the record specifications in effect at the time the record is stored. These specifications can result from default values or from overriding input from the user.

CMS records can be fixed length or variable length.

- Fixed-length records

This means that data is stored in fixed-length units. Record length can vary from file to file, but all records (lines) within a given file have the same length. CMS fills any positions of the record following the last significant character with blank characters. The user can specify the record length, but default values cover most situations. Storing data in fixed-length records can improve processing speed over variable-length records, but may not use direct access storage efficiently if a significant number of blank characters are required to fill out the records.

A Closer Look at CMS

- Variable-length records

Variable-length records can be of any length up to a maximum default value or user-specified value. CMS stores only the positions up to the last significant character. Even though overhead characters, such as line sequence numbers, take additional space, this usually results in more efficient use of direct access storage space than fixed-length records. However, variable-length record files may require more processing time.

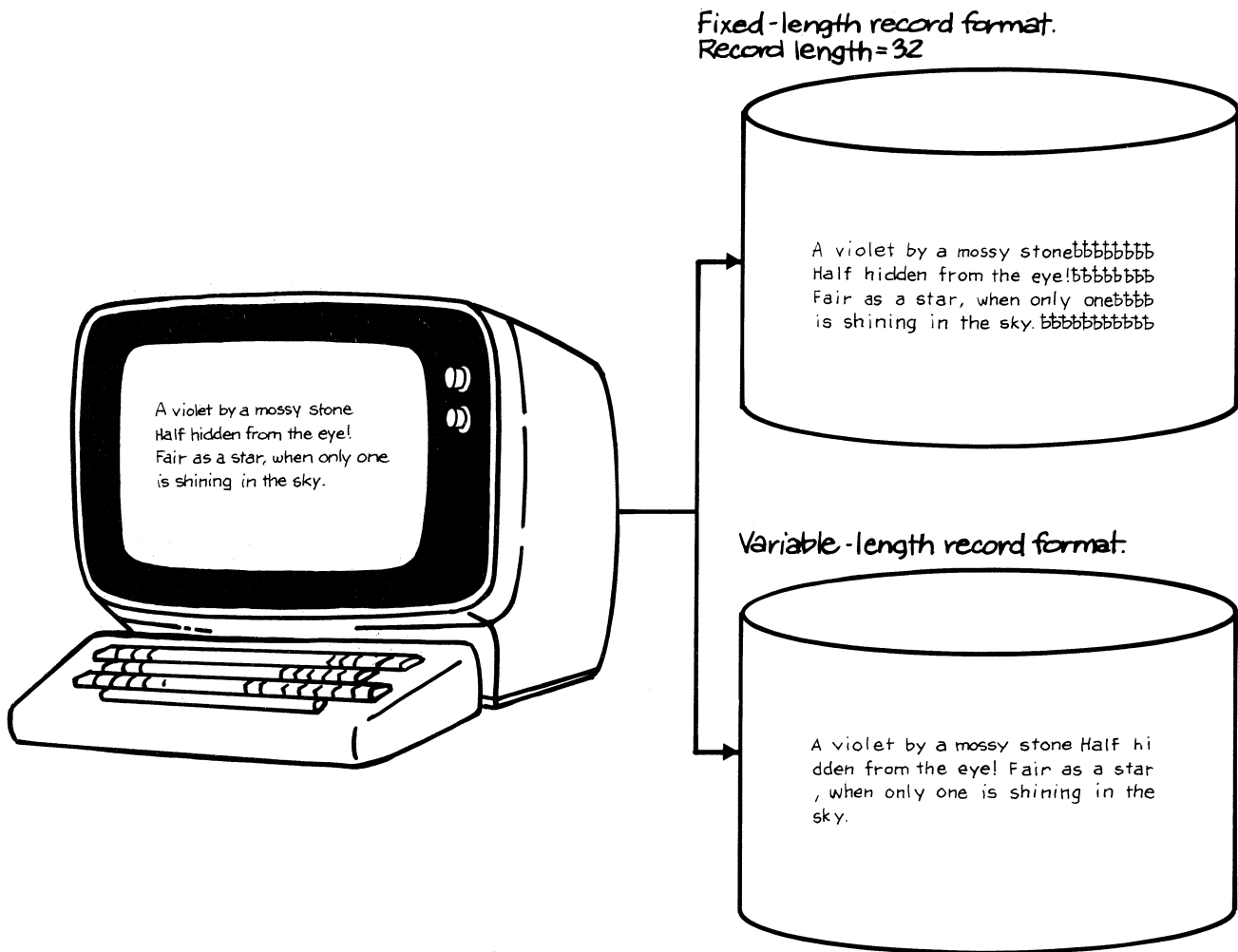


Figure 25. Fixed- and Variable-Length Records

The differences between fixed-length and variable-length records are illustrated in Figure 25.⁶ The characters at the end of each line in the

⁶ Displayed text from *Lucy*, by William Wordsworth, 1770-1850.

fixed-length format file have been inserted by the system so that each record contains a total of 32 characters (the specified record length).

It is possible for certain records to be stored more efficiently as fixed-length files than as variable-length files. This is the case when all positions of most input lines are filled with significant characters (Assembler programs for example).

Factors such as filetype (LISTING, EXEC, etc.) affect whether a file should be fixed or variable-length, and what the length should be. The System Product Editor supplies default formatting values, depending on the type of file being processed. For example, the record format and logical record length values supplied for a COBOL filetype are:

```
RECFM=F (fixed)
LRECL=80
```

For an EXEC file these values are:

```
RECFM=V (variable)
LRECL=130
```

These values for a file can be changed during an editor session, using the SET subcommand.

CMS files reside on virtual disk space that must be formatted. Virtual disk space is subdivided into blocks. A block can contain a variable number of records, depending on the block size and the record length used.

The maximum number of data items (for example, records or files) that can reside on a minidisk depends on several factors, such as the type of direct access volume used and the size of the items being stored. However, the number of data items on a minidisk cannot exceed $2^{31}-1$. That is the largest number that can be expressed in binary by the one-word (32-bit) value used by the CMS file system.

How CMS Files Are Identified

The name of every CMS file is made up of three identifiers:

- Filename - distinguishes one CMS file from another.
- Filetype - groups CMS files according to some common characteristic. This can be very helpful when using CMS facilities, such as the LISTFILE command, which allows users to selectively display only those files with common identifiers.
- Filemode - (A-Z) tells CMS on which virtual disk the file resides. A second filemode character (0-6) can be used to assign certain use characteristics to the file.

A Closer Look at CMS

CMS assigns file identifiers according to operands (or default values) for the command used to initiate the file creation. Initial file identifiers can be assigned when issuing the XEDIT command to invoke the System Product Editor. For example, issuing:

```
XEDIT BIGFILE TAXES A
```

when creating a new file causes CMS to assign the created file the identifiers:

```
Filename = BIGFILE  
Filetype = TAXES  
Filemode = A
```

File identifiers can be changed using the RENAME command under CMS. They can be changed during an editor session using the SET (FNAME, FTYPE, or FMODE) subcommand.

Filenames and filetypes can be one to eight characters long, consisting of the characters A-Z, 0-9, \$, #, @, +, - (hyphen), : (colon), or _ (underscore). Some reserved filetypes have special meanings to VM/SP and must be used accordingly. They are listed and explained in the *VM/SP CMS User's Guide*.

Typical VM/SP users often find that over a period of time they accumulate dozens or even hundreds of files on virtual direct access storage. To help recall the purpose of these files from their names, it is a good idea to assign meaningful file identifiers. For example, consider a file named:

```
DEDUCT TAX A1
```

The filename suggests that this file contains information about deductions. The filetype is a way of grouping this file with others related to taxes. The two characters of the filemode indicate:

- (A) - the file resides on the user's A-disk.
- (1) - this file can be both read from and written to. (This is the default filemode.)

Each CMS file on the same virtual disk must have a unique filename or filetype. Often the filetype, and usually the filemode, need not be specified when accessing the file. CMS may supply default values, depending on the working environment at the time.

How CMS Files Can Be Shared Among Users

An IBM program offering, VM File Storage Facility, allows users to share data files⁷ with other users. Files, or groups of files, are given passwords that provide security ranging from "anyone may access" to "available to

⁷ Whole mini-disks (as opposed to individual files) can be shared with other users by LINK and ACCESS commands (see the *VM/SP CMS User's Guide*.)

the owner only.” Passwords can give a user read access only or read and write access. Only one user can have write access to a file at one time. However, more than one user can have read access at one time, and one user can read a file while another user has write access to that file.

The VM File Storage Facility has commands that enable users to:

- Store and retrieve files
- Rename and erase files
- Send files to other users
- Find out what files they own, the size and number of records in each file, the date and by whom each file was last updated, who can share each file, and other file status information
- Change passwords.

Reference

VM/SP CMS User's Guide, SC19-6210, describes the CMS file system, together with special considerations for using it.

Virtual Machine/File Storage Facility Program Description/ Operations Manual, SB21-3085, tells how to install and use the VM File Storage Facility.

A Closer Look at CMS

CMS Session Services

Overview

CMS Session Services improve the usability of VM/SP on 3270 display terminals. It includes:

- Window functions for the end user
- A full-screen environment for CMS
- The CONSOLE macro for applications using 3270 I/O
- Updates to the System Product Editor (XEDIT)

Window Functions

Session Services allow users to manipulate information in much the same way that pieces of paper may be rearranged on a desk top. Users can display information, recall previous information and work without unexpected interruptions from the system.

In addition, users can scroll through data displayed in windows. Data can also be logged into a CMS file for future reference.

When using full-screen CMS, users of VM/SP will find it helpful to view messages through a MESSAGE window without leaving their current environment. Other windows, such as the STATUS and WARNING windows, display important system information.

More experienced users can tailor the system's default virtual screens and windows. Data can be written to virtual screens and viewed through windows. Windows can be positioned anywhere on the physical screen. They can be displayed on top of each other and may be overlapped.

Virtual Screens

In CMS Session Services, virtual screens maintain the information to be displayed in windows. A virtual screen is a functional simulation of a physical display screen. The characteristics of the virtual screen are defined by the user or the program without regard to the physical attributes of the hardware device. A program that displays data, such as the System Product Editor (XEDIT), does not have to know the device features or the size of the physical screen.

Full-Screen CMS

Full-screen capability for CMS is provided by Session Services. It is optional for 3270-type terminals. Users request to run in full-screen mode by entering SET FULLSCREEN ON or by putting this command in their PROFILE EXEC. Once in full-screen mode, users can enter data anywhere on the screen, even over existing data.

Full-screen mode uses CMS Session Services to define virtual screens and windows. VM output and messages can be routed into windows. The system allows users to determine if messages should be displayed automatically or if a notice should be issued that a message has arrived and is waiting to be viewed.

Other features of full-screen mode allow users to:

- Specify extended attributes for output
- Define Program Function (PF) keys.

Through CMS Session Services, interactive routines issue output one line at a time. CMS Session Services also capture and display CP command responses and asynchronous messages. Current machine "states" such as RUNNING and HOLDING are replaced by more meaningful status indications such as ENTER A COMMAND OR PRESS A PF OR PA KEY and EXECUTING A COMMAND.

Macro Support

The CONSOLE macro instruction is used to access full-screen console services. CONSOLE:

- Performs 3270 I/O operations
- Builds the Channel Command Word (CCW) or, for the CONSOLE EXCP function, executes the CCW built by the application
- Issues the DIAGNOSE code X'58' or SIO instruction
- Waits for the I/O to complete
- Checks any error status from the device.

CONSOLE allows programs, including CMS Session Services, to open "paths" to a display device. It coordinates use of the screen by indicating to an application writing to the device that another path has updated the screen last and that the screen must be reformatted. Thus, full-screen applications don't have to rewrite the entire screen each time a write occurs.

Two CONSOLE path names, FSOFF and FSON, are reserved for the window functions.

A Closer Look at CMS

When running in full-screen mode, the LINERD and LINEWRT macro should be used for performing line mode I/O and the CONSOLE macro for performing full-screen I/O. These macros are compatible with existing line mode interfaces but also exploit CMS Session Services functions such as the specification of virtual screen names. The LINERD macro reads a line of input from the terminal and the LINEWRT macro displays a line of output at the terminal.

The System Product Editor

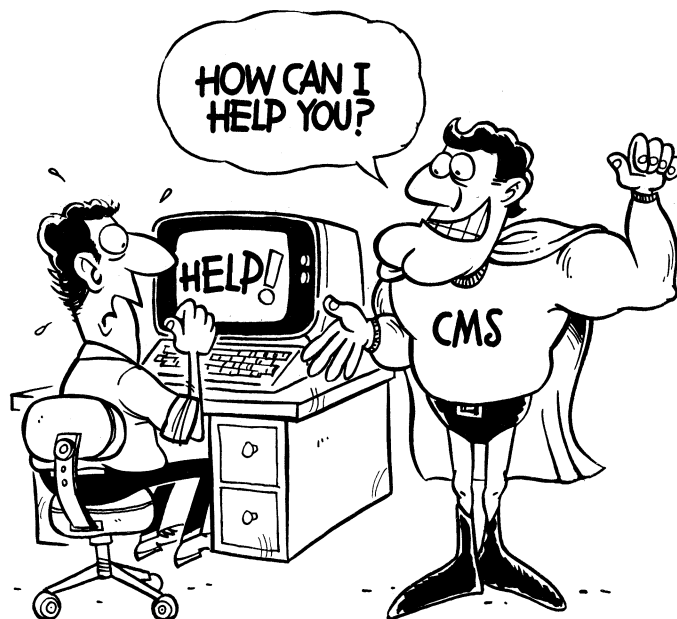
The System Product Editor (XEDIT) makes use of CMS Session Services. See the “VM/SP System Product Editor” on page 110 in this chapter.

CMS HELP Facility

Overview

HELP information is available for:

- CP, CMS, and IPCS end user commands
- CP and CMS messages
- System Product Editor (XEDIT)
- DEBUG subcommands
- System Product Interpreter (REXX)
- EXEC and EXEC2 statements
- Transparent Services Access Facility (TSAF)
- Enhanced Connectivity Facilities (SRPI)
- End user tasks.



The HELP facility is designed to be used with a 3270-type terminal in full-screen mode, but it can be used on terminals that display one line at a time. Full-screen mode refers to video display terminals that present a full screen of characters for information or editing. Line-oriented terminals display information one line at a time.

The full screens displayed by the HELP facility are called “panels.” There are three basic types of HELP panels:

- Menu panels — prompt the user to select the subject on which help is needed
- Information panels — display information about the selected subject
- Task panels — prompt the user to select the task on which help is needed.

The HELP facility is normally used to display reference information about using a particular command or subcommand, or for interpreting a system message.

To get the desired information panel displayed, a user may need to use a series of menu panels. For example, to display an information panel that tells how to use the COPYFILE command while running under CMS, a user would:

1. Issue the HELP command.
2. Follow instructions on the displayed general menu panel and select the COMMANDS panel.
3. Select COPYFILE from the displayed COMMANDS panel.

Using HELP usually takes less time than finding the information in a manual. Users can bypass menu panels by including the appropriate operand on the HELP command line. For example, issuing:

```
HELP COPYFILE
```

immediately displays a panel explaining the COPYFILE command.

The HELP command has three levels: BRIEF, DETAIL, and RELATED.

BRIEF, for the inexperienced VM/SP user, displays a command, its description, its basic syntax (command without options), an example, and, if applicable, a message telling the user that either more detailed or related information is available. If CMS Session Services is active, the information appears in a pop-up window rather than on a full screen.

DETAIL, for the experienced user, displays a complete description, syntax, and usage notes.

RELATED gives the user additional information about a command.

Requesting Help

In full screen mode, a user can request help by typing `HELP` on the command line. A user can also request help directly from menus, from the `XEDIT` environment, or from within `HELP` itself. If there is more than one panel of information, the user can scroll through it using `PF` keys.

In line mode, a user can request help by typing `HELP` and pressing the `ENTER` key.

For messages, `HELP` displays the message text, an explanation, a system action, and a user action.

Reference

VM/SP CMS User's Guide, SC19-6210, describes the CMS `HELP` facility, including how to use it or tailor it for specific needs.

A Closer Look at CMS

The Console Stack Helps Keep Work Flowing

Overview

- CMS uses a two-part storage area called console stack as a processing buffer. The two parts of the console stack are:
 - Program Stack — temporary storage for lines (or files) being exchanged by programs executing under CMS.
 - Terminal Input Buffer — holds lines keyed in from a user's terminal until CMS processes them
- CMS commands and EXEC control statements are used to manage the program stack and data flow through it.
- The sequence in which CMS normally reads input lines is:
 1. Lines from the buffer area within the program stack
 2. Lines from the terminal input buffer
 3. Lines keyed directly from the terminal keyboard.

A console stack is created in a user's virtual machine processor storage on an as-needed basis (dynamic creation). It is created to fill a specific need and varies in size according to the amount of data it must hold. Two buffer areas make up the console stack. They are:

- Program stack.
- Terminal input buffer

Exchanging Data through the Program Stack

The program stack is primarily for exchanging data among EXEC routines and application programs executing under CMS. For example, data lines can be put on the program stack using PUSH or QUEUE control statements in an EXEC routine, then retrieved with PULL control statements in another EXEC routine.

Figure 26 on page 125 illustrates the use of the terminal input buffer and the program stack.

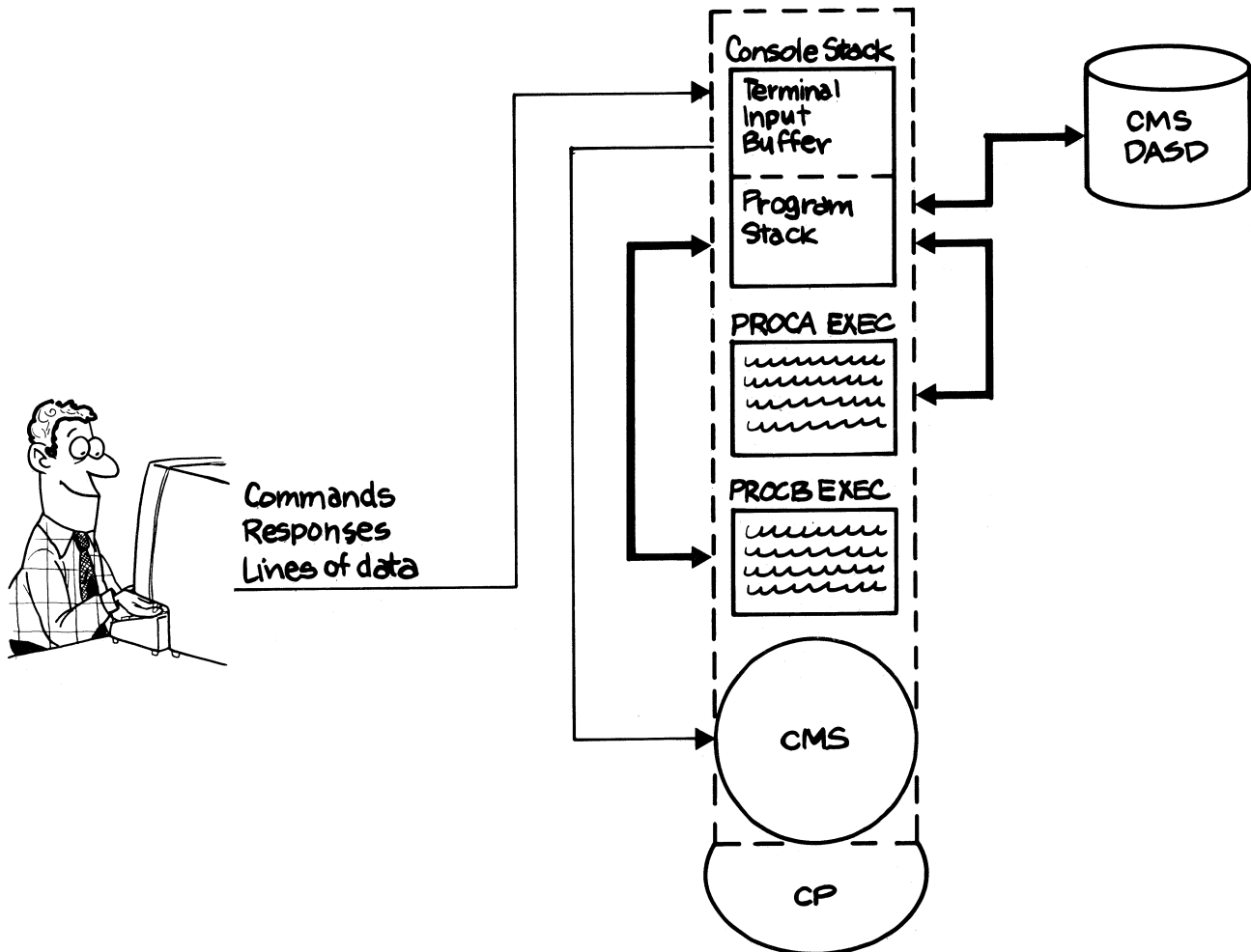


Figure 26. Using the Program Stack.

Options of the EXECIO command under CMS, using the program stack, allow users to:

- Read lines from virtual direct access storage or from a virtual reader to the program stack
- Write lines from the program stack to a CMS disk file or to a virtual unit record device
- Recover on the program stack the results of CP command execution.

Application programs running under CMS can execute the ATTN function to write a line to the program stack. To retrieve a line from the program stack, application programs execute the WAITRD function. CMS normally retrieves program stack lines in a first-in first-out (FIFO) sequence.

A Closer Look at CMS

However, the user can specify that CMS is to retrieve lines in a last-in first-out (LIFO) sequence.

CMS expands the program stack as needed from available free storage. The maximum length for lines written to the program stack is 255 characters.

Terminal Input Buffer Is Temporary Storage for Keyed Data

The terminal input buffer receives and holds lines keyed from a terminal until CMS is ready to process them. This allows a user to key in commands, responses, and other data at a rate that tends to be independent of CMS's rate of processing the input. CMS is fast, but it is often busy at the instant a user chooses to enter data. If so, the keyed lines are placed into the next available positions of the terminal input buffer. When it is ready to process more data, CMS retrieves lines from this buffer on a first-in first-out (FIFO) basis.

The maximum length for lines passing through the terminal input buffer is 130 characters.

Reference

VM/SP CMS Command Reference, SC19-6209, describes EXEC control statements and CMS application program functions available for using the program stack, including a description and examples of using the EXECIO command.

VM/SP System Product Interpreter User's Guide, SC24-5238, discusses and gives examples for using the REXX language.

VM/SP CMS User's Guide, SC19-6210, contains introductory information about writing EXECs.

IBM Enhanced Connectivity Facilities on VM/SP

Overview

- **Enhanced Connectivity Facilities is the part of CMS that allows work stations (for example, IBM Personal Computers) to communicate with a host VM/SP system.**
- **Users can start communications between VM/SP (the host) and the work station by logging on and entering CMSSERV.**
- **Application programmers can write “server” programs for VM/SP that provide functions to work station users.**

Starting Connectivity Facilities

Many times those who use work stations (IBM Personal Computers) are unfamiliar with mainframe or host computer systems; therefore, it's important that starting communications between the work station and the host be an easy process.

With the Enhanced Connectivity Facilities program running on the work station, starting the communications between VM/SP and the work station involves just logging on to VM/SP and entering CMSSERV. It's that easy. And after communications have been started, work station users can use any of the services of Enhanced Connectivity Facilities. See “Chapter 9. Connectivity, Personal Computers, and VM/SP” on page 235.

Writing Servers

A server is a program that performs one or more service functions on behalf of a requester program. A server is a host program and a requester is usually a work station program. Servers are particularly useful in that they can provide work station users with functions that might not be available on the work station alone, such as, access to host storage, printers, data bases, and communications facilities.

Application programmers can write servers (and requesters) that will use the Server-Requester Programming Interface (SRPI). The SRPI allows servers and requesters to pass information to one another between the host and work station without knowledge of the underlying communications interfaces.

A server can be a CMS module or an SRPI macro and can be written in Assembler, REXX, or EXEC 2. Its main responsibilities are to:

- Receive and validate a request from a requester
- Provide the function that the requester needs
- Send a reply back to the requester.

A Closer Look at CMS

Why might you want to write a server? Through servers, you can provide work station users with access to any VM/SP facility without forcing them to learn everything there is to know about that facility.

Reference

Introduction to Enhanced Connectivity Facilities on VM/SP, GC23-0957, describes Connectivity Facilities in greater detail.

VM/SP CMS Command Reference, SC19-6209, contains information on how to start Enhanced Connectivity Facilities using the CMSSERV command.

VM/SP Programmer's Guide to the Server-Requester Programming Interface for VM/SP, SC24-5291, provides information on how to write and install servers.

Chapter 5. Extending VM/SP's Capability

This chapter discusses VM/SP components and licensed programs that offer additional system facilities to VM/SP users.

The term “licensed program” as used in this book means a separately orderable product covered by a license agreement. Most of the licensed programs discussed in this book are also called “program products.” VM/SP itself is a program product.

The licensed programs described in this book are designed to run under VM/SP. Each one has its own documentation. The information in this chapter is to:

- Explain some VM/SP components
- Make you aware that the licensed programs exist
- Present a brief overview of the most commonly used of these programs
- Provide references to additional information.

Chapter Contents

Transparent Services Access Facility (TSAF)	131
APPC/VM Program Interface	133
Invoking the TSAF Virtual Machine	133
TSAF Commands	133
TSAF Collections	133
Group Control System (GCS)	135
GCS and SNA Networks	135
Virtual Machine Groups	136
Starting a GCS Session	136
GCS Commands	136
Network Products That Run on the Group Control System	137
NetView	138
ACF VTAM for VM/SP	138
ACF/NCP	139
ACF/SSP	139
RSCS Networking	141
The Network	142
RSCS Networking from the Operator's Point of View	143
VM/Interactive Productivity Facility	144
VM/Directory Maintenance	147
Directory Maintenance Facility System Requirements	148
Directory Maintenance from the User's Point of View	149
VM/Pass-Through Facility	150
The Pass-Through Environment	151
Pass-Through from the User's Point of View	151
Structured Query Language/Data System (SQL/DS)	154
The Data Base Structure	155
SQL/Data System from the User's Point of View	155
ISQL Allows Use of SQL/DS Interactively	156
Data Security	156
VM/SP High Performance Option	157
Improved System Management	158
Additional Processor Support	159
Additional I/O Support	160
Improved Performance and Function for MVS/SP Guests	160
Reference	161

Transparent Services Access Facility (TSAF)

Overview

The Transparent Services Access Facility (TSAF) provides access to server resources within a VM processor or across VM processors. User Access to these resources within a collection of VM processors is transparent.

TSAF includes the following:

- *Advanced Program-to-Program Communication/VM (APPC/VM)*, which is an application program interface (API) for communicating between programs in two virtual machines. This APPC/VM interface provides a limited set of the SNA LU 6.2 base communication functions. Along with the TSAF virtual machine, APPC/VM provides this communication within a single system and throughout a collection of systems.

- The TSAF function, which consists of:

- The *TSAF virtual machine component*, which is a component in VM/SP that handles communication between systems by letting APPC/VM paths span more than one VM system. Two virtual machines can communicate through programmed sequences of APPCVM and IUCV macro functions.

The TSAF virtual machine, with APPC/VM, provides the ability for virtual machines to connect to a program by specifying a name that the program has made known, instead of specifying a userid and nodeid.

- Two associated CP System Services, *Collection Resource Management System Service (*CRM)* and *Identify System Service (*IDENT)*.

Definitions you should know before reading this topic are:

- **Resource**— A program, a data file, a specific set of files, a device or any other entity or set of entities you might want to uniquely identify for purposes of application program processing in a VM system. A resource can be identified by up to eight characters.
- **Collection**— A group of VM/SP systems that can share resources. Each system within the collection must have the TSAF virtual machine installed and running.
- **Server**— A program or set of programs executing in a virtual machine and managing access to one or more VM resources; also called a resource manager.

Extending VM/SP's Capability

How TSAF Enhances Your System

TSAF provides you with the following benefits:

- **MORE FUNCTIONS AND DATA ARE AVAILABLE TO YOUR SYSTEM.**

Because TSAF lets users have access to server virtual machines on their own system and with other systems within the TSAF collection, users can access more information. Multiple users can have access to a single server virtual machine at the same time. Each system in a TSAF collection can have up to 200 resources defined.

- **TSAF APPLICATION PROGRAMS ARE EASY TO PROGRAM.**

Because TSAF lets applications connect to server virtual machines at local or remote systems within a collection by a resource name, rather than a userid, the application can reference the resource without knowing where it is located. This makes it easier for the application programmer to write APPC/VM programs. This also makes it easy to move the actual resource to another server virtual machine within the collection without changing the applications that access it.

- **THE TSAF VIRTUAL MACHINE AND THE TSAF COLLECTIONS ARE EASY TO OPERATE**

The TSAF virtual machine is easy to operate. When TSAF is started on a system, it automatically initiates communications with any other TSAF virtual machines it can reach over its defined links. The TSAF virtual machines dynamically compute the routes for their collections without the need for an operator or administrator. When a link becomes inoperative or another error condition occurs that affects the operation of the TSAF collection, the TSAF virtual machines that can still communicate with each other will reconfigure the collection. Because the TSAF virtual machines are self-configuring, they do not require much operator intervention.

Some Basics about APPC/VM

Advanced Program-to-Program Communication/VM (APPC/VM) is an application program interface (API) that provides services within a single system and, with the TSAF virtual machine component, throughout a collection of systems. To use the APPC/VM services, each virtual machine must be authorized in the system directory using the IUCV directory control statement.

APPC/VM Program Interface

The APPCVM interface, through the APPCVM and IUCV macros, provides functions to:

- Establish and sever communication paths
- Send and receive data
- Send and receive error and control information.

With a communication path authorized in the system directory, two virtual machines can communicate through programmed sequences of APPC/VM and IUCV functions.

Some Basics about the TSAF Virtual Machine Component

The TSAF virtual machine is a separate component in VM/SP that handles communication between systems by letting APPC/VM paths span more than one system. Each system has its own TSAF virtual machine.

Invoking the TSAF Virtual Machine

To start TSAF, the operator must be in the CMS environment and issue the RUNTSAF command. For a complete description of the RUNTSAF command and the sequence of events that follow, please refer to the *VM/SP Transparent Services Access Facility Reference*.

TSAF Commands

TSAF has a small set of commands. Refer to the *VM/SP Transparent Services Access Facility Reference* for a complete listing and explanation of how to use these commands.

TSAF Collections

A group of interconnected VM/SP systems that each have the TSAF virtual machine installed and running can form what is known as a TSAF collection. Virtual machines can share data within a collection.

The systems that make up the collection are connected, directly or indirectly, by either of the following TSAF controlled links:

- Channel-to-channel links (CTC), including 3088 links
- Binary Synchronous Communications (BSC) links
- 9370 Processor IBM Token-Ring Adapter
- 9370 Processor IEEE 802.3 LAN Adapter.

Extending VM/SP's Capability

Some Basics about the *CRM and *IDENT System Services

The *CRM (Collection Resource Management) System Service gives a TSAF virtual machine the ability to be a TSAF virtual machine and to query and change the local resource table. The main purpose of *CRM is to provide a communications path between CP and the TSAF virtual machine.

The *IDENT (Identify) System Service allows authorized virtual machines to be resource managers and to identify or revoke resources. The main purpose of *IDENT is to maintain a local system resource table.

Reference

VM/SP Planning Guide and Reference, SC19-6201, describes the IUCV system directory control statements.

VM/SP Transparent Services Access Facility Reference, SC24-5287, provides an in-depth description of APPC/VM, the Transparent Services Access Facility (TSAF) virtual machine component, and the *CRM and *IDENT System Services.

SNA Transaction Programmer's Reference Manual for LU Type 6.2, GC30-3084, describes the SNA LU 6.2 APPC facility.

Group Control System (GCS)

Overview

- Group Control System (GCS) is a virtual machine supervisor. It manages subsystems that support a Systems Network Architecture (SNA) network.
- Systems Network Architecture (SNA) is a description of the rules and formats for sending data through a network. SNA networks offer many benefits.
- Virtual machines in a group under GCS can share common read/write storage and communicate with each other.
- GCS can manage the many tasks of each virtual machine.
- The IPL command starts a GCS session.
- Users enter GCS commands to request system information, start work, interrupt work, and get or discard system resources.
- Users can submit programs (EXECs) to run in GCS.

GCS and SNA Networks

Group Control System (GCS) is a virtual machine supervisor that supports subsystems. One of these subsystems is Advanced Communications Function for VTAM (ACF/VTAM) Version 3 for VM/SP. Through ACF/VTAM, other programs running under GCS support a Systems Network Architecture (SNA) network. These programs and ACF/VTAM are discussed in another section of this chapter.

GCS is included in VM/SP, but users may or may not choose to install it when they generate the system.

Systems Network Architecture (SNA) is a description of the rules and formats for sending data through a network. SNA lets devices in the network do work previously done by the main processor. For instance, devices can manage communication lines, control other devices, format data, and, in some cases, execute application programs. SNA networks can benefit an installation by:

- Cutting communication line costs by using teleprocessing facilities more efficiently
- Sharing network resources

Extending VM/SP's Capability

- Allowing a flexible network configuration
- Making system growth easier to plan and manage.

Virtual Machine Groups

One or more virtual machines running under one GCS multitasking supervisor is called a Virtual Machine Group. Each virtual machine running in GCS is part of a group. Virtual machines within a group can communicate with each other and can share common read/write storage.

There can be many Virtual Machine Groups in GCS. Each group has its own GCS supervisor, and virtual machines in one group do not share storage with virtual machines in another group.

Starting a GCS Session

To start a GCS session, the operator of a virtual machine enters the IPL command. The operator enters this command using the name of the GCS system that was assigned when GCS was installed. Another way to IPL the system is to make a directory control statement entry for the virtual machine. This will automatically IPL GCS when the virtual machine logs on to VM/SP.

The system looks on the virtual machine's primary disk (A-disk) for a file named PROFILE GCS. This is like the PROFILE EXEC file in CMS. If one exists, it is executed as part of the IPL process. PROFILE GCS might contain commands that start networking application programs.

GCS Commands

GCS has a small set of commands. Users enter GCS commands to request system information, start work, interrupt work, and get or discard system resources. Programs that run under GCS may have their own commands. The *VM/SP Group Control System Command and Macro Reference* lists these commands and explains how to use them.

Reference

VM/SP Group Control System Command and Macro Reference, SC24-5250, explains the function and use of all the GCS macros.

Network Products That Run on the Group Control System

Overview

NetView is a network management licensed program. Running under VTAM, NetView can be used to automate network operations, troubleshoot network problems, and gather network management data. It includes functions of the network management products, Network Communication Control Facility (NCCF), Network Logical Data Manager (NLDM) and Network Problem Determination Application (NPDA), and functions of the program offerings VTAM Node Control Application (VNCA) and Network Management Productivity Facility (NMPF). NetView provides a cohesive set of SNA host network management services in a single product. It includes improved flow between the displays of the product components, an on-line "help desk" facility, and a browse facility. Support is provided for problem determination of the IBM 3720 Communication Controller and for on-line configuration control and testing of the IBM 586X modems. The primary objective of NetView is to enhance the usability, operability, and installability of the network management functions.

In addition to NetView, the following products will also run on GCS:

- ACF/VTAM for VM/SP is an access method that lets a VM/SP installation use the SNA network. ACF/VTAM runs under the control of the Group Control System (GCS) in its own virtual machine under VM/SP.
- VM SNA Console Support (VSCS), a component of ACF/VTAM, is an application program that lets terminals in the SNA network be used as virtual machine consoles.
- Remote Spooling Communications Subsystem (RSCS) Networking is an application program that handles the exchange of data between virtual machines in the SNA network. This is described in a separate section in this chapter.
- Advanced Communications Function/Network Control Program (ACF/NCP) runs in a 3705 or 3725 communication controller. It controls and services the devices attached to it, transmitting data between them and other parts of the network.
- Advanced Communications Function/System Support Programs (ACF/SSP) runs under CMS and provides utility functions for 3705 and 3725 communication controllers.

Extending VM/SP's Capability

NetView

NetView includes:

- **Command facility:** This facility can be used control, record, and automate various operator tasks. It can be used as an operator's interface to VTAM in a data communication network. It provides optional logging of operator commands and messages on either disk or printers. It also supports the programmable operator facility of VM permitting remote operations.
- **Session monitor:** The session monitor can be used to collect, correlate, and display data about sessions and routes. It allows you, from a central control point, to examine information related to the SNA network and to identify network problems.
- **Hardware monitor:** The hardware monitor gathers and displays problem determination information. This information is generated at resources that are link- or channel-attached to the host system.
- **Status monitor:** The status monitor provides a full-screen means of controlling a network. It displays network status and accepts network operator commands. It also allows for automatic reactivation of failed nodes.
- **Browse function:** This function allows the operator to browse library files, such as VTAM major nodes or NetView definitions. The operator can also browse a log of all operator commands and messages.
- **Online help:** NetView includes help panels that explain the syntax and usage of each command. Explanations are also provided for recommended actions, VTAM states, and VTAM return/feedback codes.
- **Online help desk:** The help desk provides step-by-step instructions for operators performing network problem determination. Users can customize the help desk to conform to their own configurations and procedures.

ACF VTAM for VM/SP

Advanced Communications Function for the Virtual Telecommunications Access Method (ACF/VTAM) for VM/SP lets an installation implement SNA in a VM/SP environment. It runs under the control of the Group Control System (GCS) in its own VM/SP virtual machine. ACF/VTAM controls data flow between the devices defined to it and ACF/VTAM programs running in other virtual machines.

All terminals, programs, lines, and devices defined to ACF/VTAM become part of the SNA network. Existing devices and programs that cannot or should not be part of the SNA network can coexist with it. With ACF/VTAM, an installation no longer needs a guest OS/VS or VSE operating system to get SNA support.

VM SNA Console Support (VSCS) is a component of ACF/VTAM. VSCS is based on a program product, VM/VTAM Communications Networking Application (VM/VCNA), which it replaces. VSCS is an ACF/VTAM application program that lets terminals in the SNA network be used as virtual machine consoles. It does this by communicating, via the Advanced Program-to-Program Communication (APPC), with the Console Communication Service (CCS) component of CP. Even though VSCS comes with ACF/VTAM, it can be started and stopped independently, it offers separate trace and dump facilities, and it can run in its own virtual machine.

ACF/VTAM supports or is used with program products that help manage communication networks. These products are listed and explained below.

ACF/NCP

Advanced Communications Function/Network Control Program (ACF/NCP) manages many functions in the network, including:

- Sending data to other parts of the network
- Controlling line traffic and buffering
- Deleting and inserting communication control characters
- Gathering line statistics
- Starting and shutting down lines
- Closing down parts of the network
- Handling minor line errors
- Giving error statistics to ACF/VTAM
- Testing connections to other network elements.

ACF/SSP

Advanced Communications Function/System Support Programs (ACF/SSP) provides utility functions for 3705 and 3725 communication controllers. It loads the Network Control Program (NCP) into a controller, starts an NCP dump of the controller, and generates an NCP.

Extending VM/SP's Capability

Reference

Network Program Products General Information, GC23-0108, has general information about VTAM and programs that run under it.

Network Program Products Planning, SC23-0110, contains planning information for VTAM and the programs that run under it.

RSCS Networking

Overview

- **The RSCS Networking Version 2 program product manages data transfer among systems spread over multiple sites (distributed sites). It runs on the Group Control System part of VM/SP and enables virtual machine users to:**
 - Transmit and receive commands and messages
 - Transmit files to remote users as requested by the sending user
 - Receive files from remote users on behalf of the receiving user
 - Print output (including graphics) on printers it controls
 - Transmit and receive output for system controlled devices.
- **RSCS Networking provides for remote job entry (RJE) into a VM/SP system.**
- **Each RSCS Networking virtual machine directory defines the local system and its view of a network of systems.**

Definitions that you should know before reading this topic are:

- **Network** — A distributed data processing environment in which the locations are interconnected.
- **Link** — Facilities for communicating between two adjacent nodes in a network.
- **Node** — A defined point in a network where one or more links terminate.

The Remote Spooling Communications Subsystem (RSCS) Networking Version 2 program product is a single-purpose operating system. Its purpose is to handle the exchange of data between virtual machines in a network of VM/SP systems. This includes the printing of files on remote IBM 3270 Information Display System (or equivalent) printers in an RSCS Networking network.

RSCS Networking Version 1 can still be used in those installations where the SNA network is not needed. However, RSCS Networking Version 2 can be used in both the SNA and a non-SNA environment.

Extending VM/SP's Capability

To transmit data to a user on another system, the sender must tell RSCS Networking the location and userid of the receiving user and the name of the file to be sent. From the CMS user's point of view, this can be easily accomplished through use of the SENDFILE command (Figure 27 on page 142).

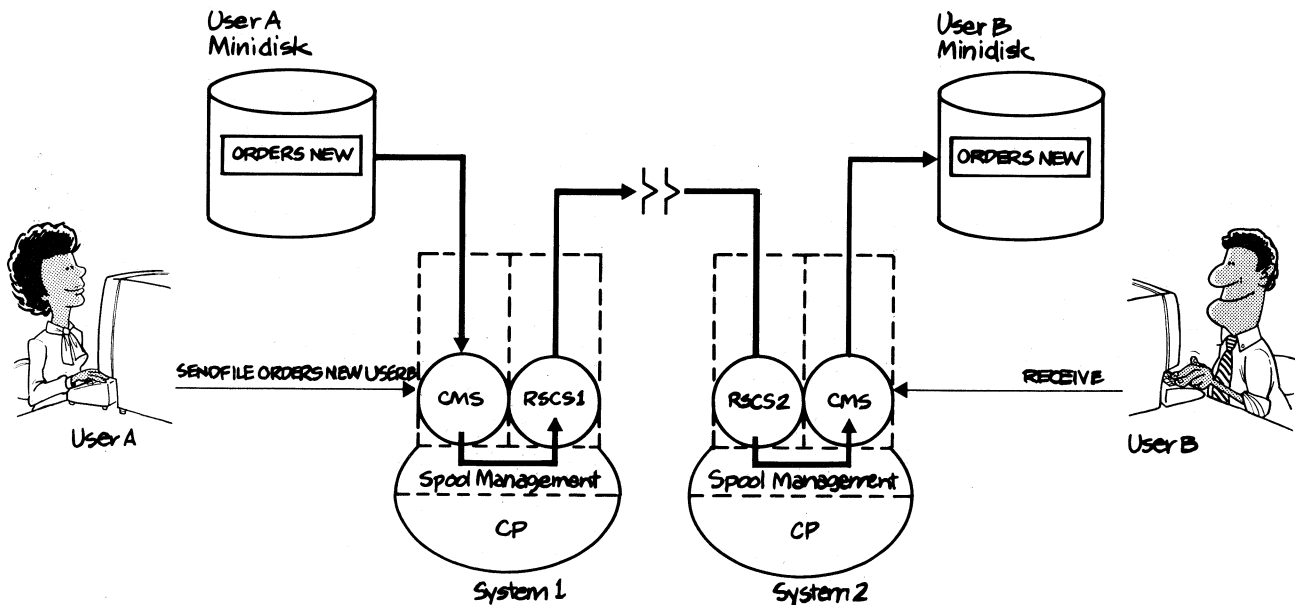


Figure 27. How RSCS Networking Transmits a File to a Remote Non-SNA node

The Network

RSCS Networking runs on the Group Control System (GCS). As also shown in Figure 27, it runs in a virtual machine on each of the VM/SP systems that make up a communications network. Each of these RSCS Networking operating systems has a directory of data that defines the network. Directory entries include such things as:

- The identification of the local node (nodeid)
- The identification of each network node with which the local node can communicate and whether the node is SNA
- Network paths (links) over which the local node communicates with other nodes in the network.

This data enables each RSCS Networking system to decide how to handle data entering or leaving its node.

Data transmission can be over long-distance communications facilities or, for systems at the same location, over channel-to-channel adapters. Each path between two systems that data can travel, without going through an

intermediate system, is called a link. Each point in the network where links terminate is called a node.

RSCS Networking from the Operator's Point of View

RSCS Networking has its own set of commands and messages to help the operator manage network operation. The operator's RSCS Networking tasks include:

- Starting (IPLing) and stopping RSCS Networking virtual machine activity
- Managing network links between the local node and other nodes
- Managing transmission of files passing through the local node
- Managing the printing of output on printers it controls
- Transmitting commands and messages
- Monitoring network activity.

Reference

RSCS Networking Version 2 General Information, GH24-5055, describes the program and other books in the RSCS library.

Extending VM/SP's Capability

VM/Interactive Productivity Facility

Overview

- **VM/Interactive Productivity Facility (IPF) gives users a simplified and productive interface to a VM/SP system.**
- **Tasks are initiated by issuing the IPF command, then responding to a series of prompting displays (panels).**
- **IPF communicates with users through dialogs made up of four types of visual panels. They are:**
 - **Menu**
 - **Data entry**
 - **HELP**
 - **Output.**

The IPF licensed program runs in a CMS virtual machine. It allows users to do tasks under VM/SP without having to remember or look up related commands. IPF is normally invoked by issuing the DTRIPF command, either manually under CMS or as part of an automated IPL procedure. From that point a series of displayed menus allow selection of the task to be done without the user actually issuing VM/SP commands. The menus are designed to fill the needs of the type of user invoking the IPF facility.

The task selection process, using formatted menu panels, is simple. The user simply keys selections into the appropriate panel areas and presses the ENTER key. For many panels, IPF automatically positions the cursor for keying in data. For other panels, the tab key can be used to quickly position the cursor.

The Interactive System Productivity Facility (ISPF) licensed program must be installed in addition to IPF. ISPF provides control and services that support dialogs (prompt-response sequences) between users and IPF. *ISPF Dialog Management Services* is the source for information about ISPF.

Figure 28 on page 145 is a typical IPF menu panel. This panel shows users a selection of tasks to do under VM/SP. For example, to use a magnetic tape, users begin by keying in the number 5, then respond to subsequent prompts on displayed panels. To perform this operation without IPF, users must know how to request help from the system operator using VM/SP commands. Otherwise, they must find the procedure in a reference book or get help from someone who knows how to do the job.

IDM\$GNRC	GENERAL VM/SP TASKS	

To choose one of the following, type its number and press ENTER:		
1 File-manage	Manage files.	
2 Copyfile	Copy a file.	
3 Communicate	Communicate with others.	
4 Disk-access	Get a disk.	
5 Tape	Use a tape.	
6 Devices	Manage your virtual devices.	
7 Status	Query or change status.	
8 Libraries	Use libraries (Text library and Macro library).	
9 Network	Query RSCS functions.	

PF: 1 Help	2 ...	3 End
PF: 7 ...	8 ...	9 Probrep
		4 Return
		5 Cmd
		6 Index
		10 ...
		11 ...
		12 Cursor
====> _		

Figure 28. VM/IPF General Task Menu Panel

There are three groups of IPF panels, each group relating to a user type. The three panel groups are:

- General User
- System Administrator
- System Operator.

IPF uses default values for many panel entries if they are not filled in. Under IPF, program function keys are automatically set to do useful functions to make the facility easy to use. IPF displays an abbreviated description of these keys at the bottom of each panel.

IPF has its own facility for handling various kinds of problems. It is the Problem Control Facility (PCF). PCF offers a panel-assisted means of recording problems and transmitting them to the appropriate person. Problem reports can be generated and distributed within the same system, or, using facilities of the RSCS Networking licensed program, sent to other VM/SP systems with IPF installed.

Extending VM/SP's Capability

Reference

VM/IPF Operation, SC24-5229, describes the program and its use for the operator.

VM/IPF Administration Guide, SC24-5230, describes the program and its use for the system administrator.

VM/IPF General Use Guide, SC24-5233, describes the program and its use for the general user.

ISPF Dialog Management Services, SC34-2088, describes the program.

VM/Directory Maintenance

Overview

VM/Directory Maintenance program product provides for:

- **Directory update capability for both the system administrator and the general user**
- **Optional enforcement of regular password changes**
- **Updating the directory using a controlled service virtual machine to reduce the chances of errors**
- **Improved directory availability if a program interrupt or a disabled wait state occurs**
- **Prevention of allocating new minidisk space over existing minidisk extents**
- **Ability of the general users to make simple directory changes for themselves**
- **Use of HELP panels.**

Maintenance of the VM/SP system directory can be time-consuming and error prone. The VM/Directory Maintenance program product is designed to make this task easier and more efficient. Using Directory Maintenance commands, changes to the VM/SP system directory can be made interactively. The alternative, updating the directory manually, takes longer and is more likely to cause human error.

Directory Maintenance provides two classes of commands:

- General user commands
- System administrator commands.

By enabling general users to make simple updates to the directory, the necessity of doing full updates is reduced. Updating areas of the directory that require special access authorization can be restricted to administrators who have a privileged class of commands available to them for that purpose.

Extending VM/SP's Capability

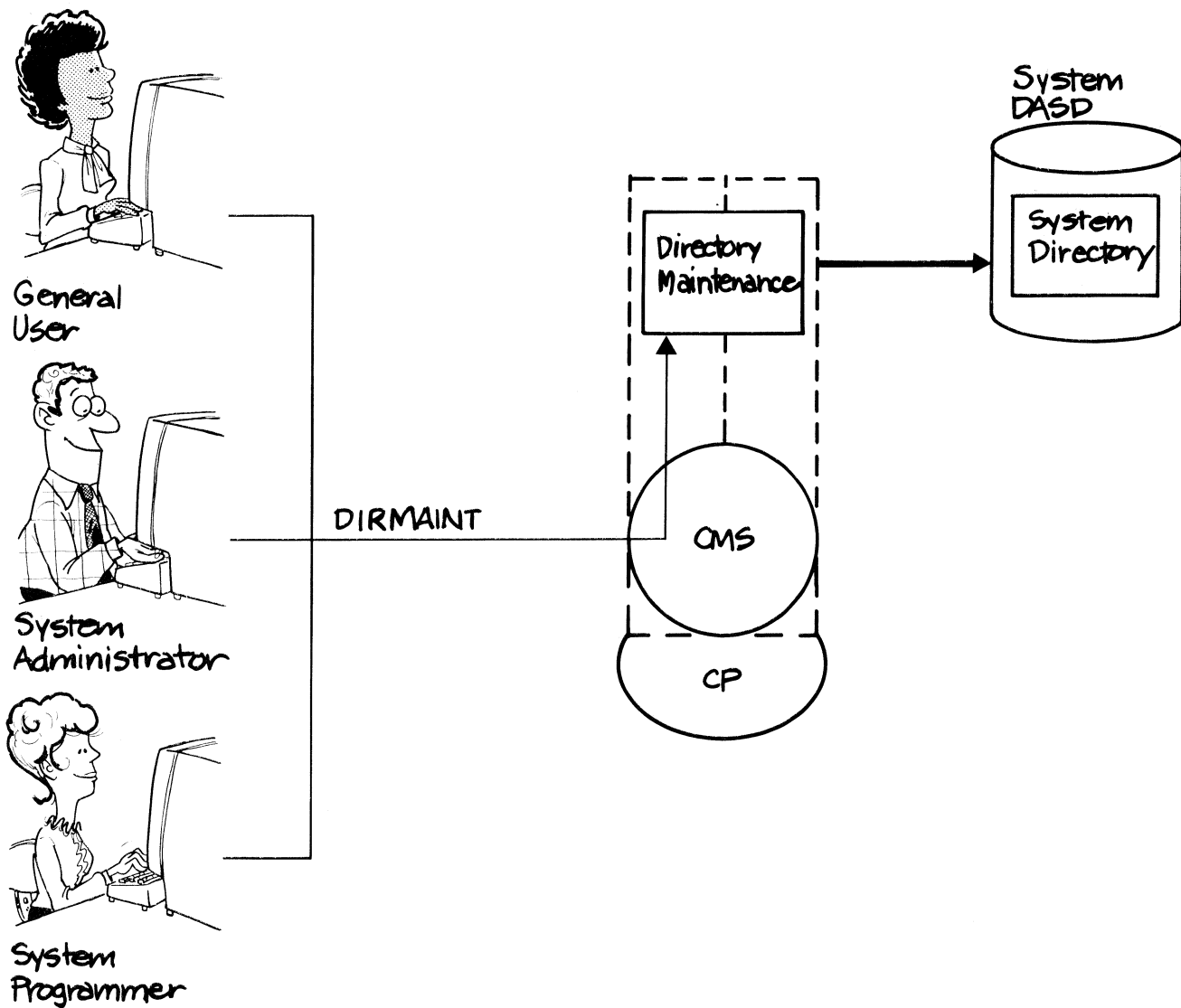


Figure 29. Updating the VM/SP System Directory

Directory Maintenance Facility System Requirements

Using Directory Maintenance requires two virtual machines:

- DIRMAINT - is the primary directory manager.
- DATAMOVE - does copying and formatting functions.

These virtual machines normally run in a disconnected state.

Directory Maintenance requires three minidisks:

- A-disk - contains the current directory source file, a history file, and control files.
- B-disk - contains a directory source backup file, plus a journal of changes since the last backup was created.
- C-disk - contains Directory Maintenance programs.

Directory Maintenance from the User's Point of View

Requests are communicated to the directory maintenance virtual machines using options of the DIRMAINT command under CMS (Figure 29 on page 148). When the requested function is completed, the user's virtual machine receives a completion message.

A general user is authorized to use only a subset of the DIRMAINT command options. The user can request changes that affect virtual machine characteristics, such as password changes or automatic IPL. The user cannot request changes that affect use of system resources, such as improving dispatch priority or changing virtual disk allocation. The system administrator is authorized to make directory changes that affect system resources, using DIRMAINT command options not available to the general user.

Reference

VM/Directory Maintenance General Information, GC20-1836, provides an overview of the program.

VM/Directory Maintenance Guide for General Users, SC20-1839, describes how to use the program.

VM/SP Planning Guide and Reference, SC19-6201, describes VM/SP system directory entries for Directory Maintenance.

Extending VM/SP's Capability

VM/Pass-Through Facility

Overview

- The VM/Pass-Through Facility program product (Pass-Through) allows VM users to interactively access network nodes.
- Pass-Through provides for:
 - Attachment of IBM 3270 display stations to selected nodes
 - Attachment of IBM 3270 Information Display System printers to non Pass-Through systems
 - Routing of information among distributed Pass-Through nodes.
- VM users can access the Pass-Through facility either from the CP environment or when running under CMS.

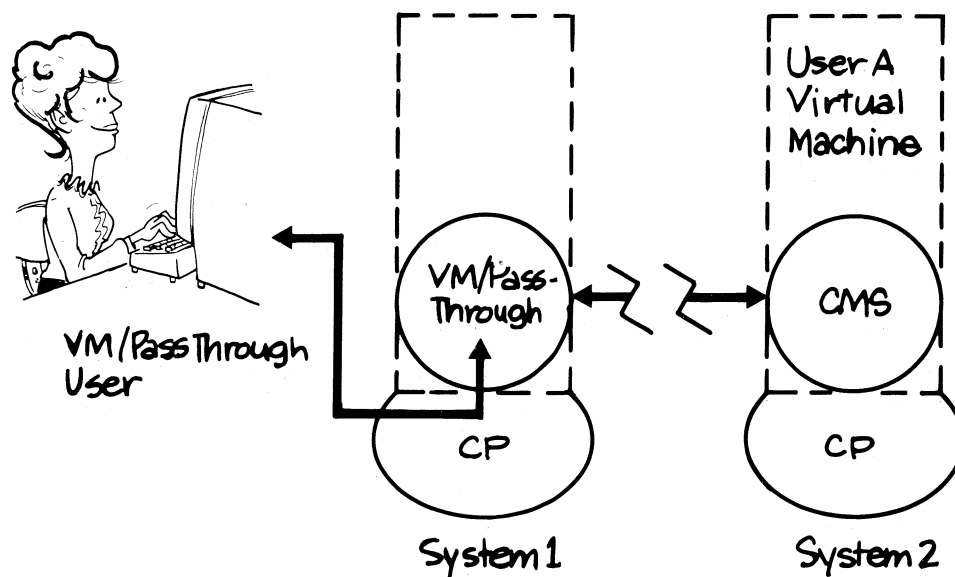


Figure 30. Using VM/Pass-Through to Access Another Network System

Pass-Through can be very useful for people who need to use systems other than the VM/SP system to which their terminal is attached. Using Pass-Through, users can log onto and use another system in a defined network as though they were at a console directly attached to that system.

The Pass-Through Environment

One or two virtual machines in the local VM/SP system (the one through which the Pass-Through session is initiated) may be involved in a Pass-Through session:

- The CMS virtual machine in which Pass-Through runs
- The user's virtual machine, if the user accesses Pass-Through by using the PASSTHRU command under CMS.

In addition, if a user logs onto another VM system in the Pass-Through network, CP in the distributed system creates a virtual machine for that user's session.

Each Pass-Through virtual machine maintains a directory called the configuration file. It contains information needed for network management, such as:

- Local and remote node identifications
- Link and routing definitions
- Authorization for using restricted commands.

Pass-Through must be installed on each VM network system on which users initiate Pass-Through sessions. It must also be installed on any intermediate nodes. Nodes are systems at which network communication links terminate. Intermediate nodes are systems through which Pass-Through data passes as it travels from the source node to a target node. However, Pass-Through need not be installed on passive systems in the network, that is, systems whose only Pass-Through function is that of being accessed by users on other systems.

The system directory of each network system must identify all users at other nodes who might access that system. From a Pass-Through point of view, every user on the network is potentially a local user.

Pass-Through from the User's Point of View

Initiating a session on a network system consists of three easy steps.

1. Issue the DIAL command to CP without logging on, or issue the PASSTHRU command from the CMS environment after logging onto the system. Either causes a menu of network systems to be displayed.
2. Select the system to be accessed as prompted on the displayed menu.
3. When the logo of the selected system is displayed on the local console, log onto the accessed system.

Because the user in Figure 30 on page 150 does not have a virtual machine on the local system (the one shown is for the Pass-Through facility), it can

Extending VM/SP's Capability

be assumed that the session was initiated using the DIAL command. In this case the user has initialized CMS as the operating system on the second system. The double-ended arrows in the figure point out that Pass-Through is used for interactive communication.

Once the logon process is completed, most Pass-Through program activity becomes transparent to the user.

When the session on the accessed system is completed, the user logs off that system. If the Pass-Through session was initiated from a CMS environment (PASSTHRU command), the Pass-Through session is terminated. If the Pass-Through session was initiated from the CP environment (DIAL command), the network menu again appears on the user's console. The user can then initiate a session with another network system or terminate the Pass-Through environment.

Pass-Through has its own set of commands and messages, some of which can be used by the general user. For example, there are general user commands to establish a Pass-Through session, query status, or send messages.

Many Pass-Through commands can be issued only from the Pass-Through virtual machine console or by persons identified by AUTHORIZ records in the local configuration file. The system operator (Pass-Through operator) uses these commands to manage the network. The operator does such things as:

- Log on and initialize Pass-Through
- Start telecommunication line drivers
- Initialize remote processors.

Some other facilities that Pass-Through offers are:

- A user can disconnect from a Pass-Through session (processing in that system continues), return to the local CMS virtual machine to do some processing, and then resume the disconnected session.
- If a Pass-Through session is initiated from the CMS environment, users can store console screen images on their virtual A-disks by pressing a program function (PF) key.
- Program exits are provided to allow users to supply session control routines, such as a routine to gather accounting data.
- An IBM 3270 Information Display System printer can be assigned to be the output device for a session on a remote system.

Reference

VM/Pass-Through Facility General Information, GC24-5206, provides an overview of the program for a potential user.

VM/Pass-Through Facility Guide and Reference, SC24-5208, provides an in-depth discussion of the program.

Structured Query Language/Data System (SQL/DS)

Overview

The Structured Query Language/Data System (SQL/DS) is a data base management system that provides:

- **Data sharing** - users can share data with each other
- **Multi-write** - users can update concurrently
- **Views** - various fields of data can be selected for viewing
- **Data security** - the owner of the data can control access to the data
- **Data recovery** - data can be recovered up to the last update in the event of power or system failures.
- **The SQL/DS program product manages structured data stored in a computer.**
- **SQL/DS manages data stored in tables.**
- **Data access is via a set of commands called Structured Query Language (SQL).**
- **SQL commands offer the user a variety of functions to define, manipulate and control access to data.**

SQL/DS is designed to be used by people with little or no data processing experience. With it a user can easily access information stored as tables. Data can be accessed by:

- issuing SQL commands from a terminal
- executing a routine containing a series of SQL commands
- embedding SQL commands in programs written in COBOL, PL/I, FORTRAN, or Assembler language
- using any one of a number of end user products that support SQL/DS data (see "Chapter 6. Applications for VM/SP" on page 163).

The Data Base Structure

Data in an SQL/DS data base is organized in table (rows and columns) format. A data base may consist of one or more tables. Three designations facilitate the location of data within these tables.

- Each of these tables has a unique name.
- Each column within a table has a unique name.
- Rows within a table can be identified by field content.

Selecting data from SQL tables requires knowing at least the names of the tables in which the data is stored. Some data bases can contain many tables, so remembering all their names is not practical. To help, SQL/DS maintains catalogs of information about the tables that make up the data base. The catalogs contain, among other things, table and column names for all tables in the data base. The SELECT command for retrieving table data under SQL/DS can be used to display information from these catalogs to help find the table and column names needed.

SQL/Data System from the User's Point of View

Working with a structured data base, SQL commands are available to:

- Select an item or group of items from the data base
- Update an item or group of items
- Use as statements in programs that use SQL/DS
- Control authorization to use the data base
- Manage the data base and its use.

SQL/DS is commonly used to display selected columns (and perhaps selected rows within those columns) of data from a data base table(s). The basic SQL query command:

```
SELECT (column names) FROM (table name) WHERE (row select conditions)
```

allows users to display data from any table in the data base.

An entire table or selected columns and/or rows from the table can be displayed. For example, suppose that a given table contains records of sales data. Information in the table is structured so that the name of the city associated with each row of data is always entered in the column named CITY. This allows users to request that all rows of that table containing BALTIMORE in the CITY column be displayed.

Extending VM/SP's Capability

Besides simple queries, a SELECT statement can:

- Contain logical operations such as “greater than”
- Request calculations, including built-in functions like maximum or average
- Select data from more than one table at a time.

If needed, a copy of the query results can be printed.

SQL/DS, as it applies specifically to the application programming environment and the business environment, is described in later topics of this book.

ISQL Allows Use of SQL/DS Interactively

SQL/DS includes a facility, Interactive SQL (ISQL), that enables users to access and manipulate data directly from a terminal. ISQL commands are used to manage a SQL/DS session interactively to do such things as:

- Reissue, with or without changing it, the previous (current) SQL command issued from that terminal
- Format query results before they are printed
- Store frequently used SQL commands for reuse without retyping
- Specify the page size and number of copies for printed reports.

Data Security

All data stored in a SQL/DS table is owned by the SQL/DS user who created the table. The table owner controls all types of access to that table. The owner can do various types of management activities on the table and can grant some or all table management privileges to specific users. These privileges can be revoked as well as granted. Data security is therefore the responsibility of the table owner.

Reference

SQL/Data System General Information for VM/SP, GH24-5064, provides an overview of the program for those who make decisions about how suitable it is for their installation.

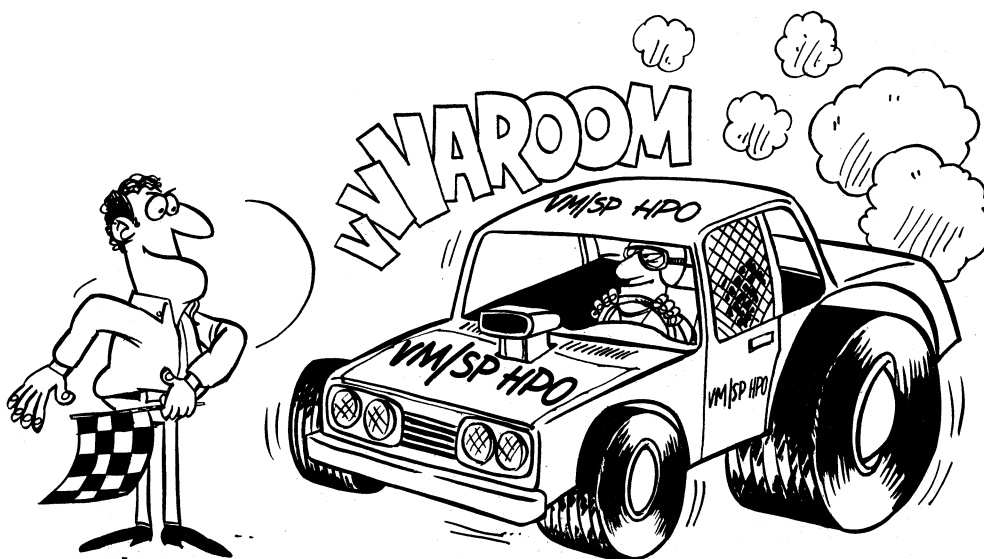
SQL/Data System Concepts and Facilities for VM/SP, GH24-5065, provides a closer look at SQL principles and their implementation.

SQL/Data System Terminal User's Guide, SH24-5016, describes how to use the program.

VM/SP High Performance Option

Overview

- The VM/System Product High Performance Option (VM/SP HPO) licensed program extends the functional capabilities of the CP component of VM/SP.
- VM/SP is a requisite program for running VM/SP HPO.



What Does VM/SP HPO Add to VM/SP?

VM/SP HPO extends the capabilities of CP in the following areas:

- Improved system management
- Additional processor support
- Additional I/O support
- Improved performance and function for MVS guests.

Extending VM/SP's Capability

Improved System Management

Segment protection extension: Segment protection support in VM/SP HPO uses the segment protection feature — a processor enhancement that uses microcode assist — to provide shared segment protection. The segment protection extension, a standard feature on the 308x, 3090, and 4381 processors, prevents altering of shared segments that are designated as protected in the NAMESYS macro instruction.

The feature improves performance for CMS shared segment users by eliminating CP monitoring of shared segment pages.

Dispatching and scheduling modifications: VM/SP HPO uses a dispatch list, a dispatcher fast path, and scheduler design changes to improve its efficiency.

The dispatch list contains only those virtual machines that are dispatchable, unlike the run list, which contains all virtual machines, some of which are not executable. The control program scheduler still maintains the run list, but searches only the dispatch list when selecting a virtual machine to run.

The dispatcher fast path allows some virtual machines, such as CMS, to be quickly redispached after events that interrupt the execution of the virtual machine but do not change its status.

The scheduler design changes maintain interactive response times under more varied system loads.

Free storage management improvements: VM/SP HPO manages free storage subpools more efficiently. Subpools are fixed-size storage groups; they are used to satisfy control program requests for free storage when CP needs to perform system-related functions such as building control blocks, processing I/O operations, or building save areas.

Enhanced paging subsystem support: This support improves paging performance by using main storage as a high-speed buffer and by grouping together user pages that are likely to be used together. These groups of user pages are swapped in and out of main storage with one start I/O (SIO) operation. In addition, the system programmer can use the SYSPAG macro to use DASD more efficiently.

Expanded Storage support: Expanded Storage is a paging assist on the 3090 processor. The control program of VM/SP HPO uses it as a high-speed paging area.

N-select: N-select is a DASD page-slot selection algorithm that allocates a selected number of consecutive requests to the same allocation area of DASD. This enables CP to write multiple pages to the same device with a single SIO.

Active Wait: In AP or MP systems, work can become available without an accompanying interrupt. When one processor becomes idle, it goes into an active wait.

Active wait lets an idle processor in an AP or MP system scan the dispatch request queues and dispatch lists looking for work.

Swapping: Swapping relates to the paging in or paging out of groups of pages. It provides lists of pages that can be moved to paging media when page frames are needed. Swapping also has the following objectives:

- It tries to keep the working sets of interactive users in real storage across queue drops.
- It provides more efficient paging media use when page out is needed.

SET RESERVE for Multiple Users: The system operator can use the SET RESERVE command to reserve real storage for multiple virtual machines. When a virtual machine has this option, during its execution CP dynamically builds a set of reserved page frames until the maximum number "reserved" is reached.

The SET RESERVE command can increase the efficiency of certain noninteractive virtual machines such as system control programs and special service machines.

Vector Facility: The Vector Facility is an instruction processor capable of high-speed manipulation of values, usually floating-point values. VM/SP HPO supports the Vector Facility in System/370 mode when it is available on a processor. A Vector Facility can be attached to only one processor, but each processor in a processor complex can have a Vector Facility.

Capability for 9900 Spool Files per User: An installation can support up to 9900 spool files per user subject to limited checkpoint and storage space. Recovery of spool files during checkpoint and force starts is redesigned by starting multiple tasks that overlap both processing and I/O activity.

Additional Processor Support

Extended storage protection key support: This feature enables CP to initialize itself on processors that have real storage frames protected by a single 4K storage protection key rather than two 2K keys.

Extended storage support: This feature allows CP in VM/SP HPO to manage processor storage that exceeds 16 Mb. CP uses the storage above 16 Mb for virtual machine pages. This can result in improved productivity for systems that were previously constrained by limited availability of real storage.

Extending VM/SP's Capability

Dyadic and dual processors support: The performance on dyadic and dual processors with store-in caches is improved by implementing control block alignment, local dispatcher and free-storage queues for each processor, and dispatcher queue scanning.

- Frequently used control blocks are aligned on cache-line boundaries.
- Dispatcher and free-storage queues are provided for each processor. Thus two processors will modify and reference the same storage areas less frequently.
- When a processor completes its dispatchable work, it scans the dispatcher queues of the other processor instead of going into a wait state. The other processor no longer has to signal the idle processor when there is a piece of work that the idle processor can perform.

Additional I/O Support

IBM 3880 Model 11 and 21 storage subsystems support: Excessive paging demands can cause active pages to be transferred from real storage to auxiliary storage. This support keeps frequently used pages more readily available to the system, by transferring active pages between real storage and a high-speed buffer area called the cache.

IBM 3880 Model 13 and 23 storage subsystems support: This support is designed especially for nonpaging applications such as MVS or CMS guest virtual machines running with 3380 DASDs. The performance improvement comes by maximizing the number of read accesses that can be satisfied by accessing the cache (subsystem storage) copy.

Extended Channel support: This enables an installation to configure up to 24 or 32 channels in a 3090 uniprocessor configuration, up to 48 channels in a 3090 dyadic configuration, and up to 96 channels in a 3090 partitioned processing configuration.

Extended virtual device support: This extends the number of virtual devices that you can support. The exact number of virtual devices that you can define depends on the release of VM/SP HPO that you are using.

Improved Performance and Function for MVS/SP Guests

Preferred machine assist support: VM/SP HPO supports the preferred machine assist feature, a microcode processor enhancement that is standard on the 308x, and 3090, and 4381 processors, and is separately orderable for the 3033. Preferred machine assist improves performance for the MVS/SP V=R preferred guest by reducing control program processing.

As a preferred guest, MVS/SP operates in supervisor state with direct control of its hardware resources and I/O operations. The preferred guest can also use storage greater than 16 Mb. This provides even more performance potential for storage-constrained MVS/SP systems.

Preferred machine assist also removes restrictions for using single processor mode.

Control switch assist extensions to preferred machine assist: This support, available on 308x, 3090, and 4381 processors, allows the MVS/SP preferred guest to use IUCV, some DIAGNOSE instructions, and some Service Call instructions. It also reduces line timeout problems for such guests by letting CP reflect virtual I/O interruptions to the guest. You must install the control switch assist to obtain this support.

Enhanced availability in the MVS/SP V=R environment: VM/SP HPO attempts to save the status of the V=R (virtual addresses = real addresses) virtual machine in which it is running, if an abnormal termination of CP occurs. When CP is automatically restarted, execution in the MVS/SP V=R virtual machine resumes. When the MVS/SP virtual machine is operating with preferred machine assist active, additional recovery capability is provided.

3033 Extension Feature (Virtual Machine Assist enhancement): This enhancement improves performance by reducing paging operations, improving real storage management, and increasing the performance of cross-memory services and page fault processing.

Single processor mode operational enhancements: Single processor mode lets you restrict the VM/SP HPO control program to a single processor in an attached processor or multiprocessor system, leaving the other processor for the exclusive use of the MVS/SP virtual machine. These enhancements make it easy to switch between single processor mode and either attached processor mode or multiprocessor mode.

Reference

The VM/SP High Performance Option licensed program is documented in its own library that parallels the VM/SP library.

The *VM/SP High Performance Option Release 5.0 Guide*, SC23-0189, describes in more detail how VM/SP HPO extends the capabilities of VM/SP.

Extending VM/SP's Capability

Chapter 6. Applications for VM/SP

This chapter discusses licensed programs designed to provide additional application capability for VM/SP users. These programs run under CMS. Although not all licensed programs that run on VM/SP are discussed, those included provide a good sample of the different types of programs that are available.

Additional sources of information about programs that run on VM/SP are:

- *Software Directory*, GB21-9949
- *The Software Catalog*, G320-6531
- *IBM Engineering and Scientific Application Programs Available From Non-IBM Sources*, G320-6739
- *VM Application Programs Available from Non-IBM Sources*, GY33-6951.

Chapter Contents

Application System (AS)	165
Business Planning	165
Statistics and Forecasting	165
Project Control	166
Document Preparation	166
Business Graphics	166
Professional Office System (PROFS)	168
Preparation of Documents	169
Distributing and Receiving Documents	169
Formatting and Printing Documents	169
Retrieving Documents	170
Handling Routine Office Tasks	170
Document Composition Facility (DCF)	171
SCRIPT/VS Is for Detailed Format Markup	172
GML Is High-Level Format Markup Language	173
Processing DCF Formatted Documents	174
DisplayWrite/370	175
Full-Screen Support	176
Extended Attributes	176
Document Representation	176
Cursor Movement	176
Window Techniques	176
Error recovery	177
Help Facility	177
Tailoring DisplayWrite/370	177
Split Screen	177
Linguistic Support	178
DisplayWrite/370 Language Dictionaries	178
National Language Support	179
Graphical Data Display Manager (GDDM)	180
GDDM Utilities for Drawing Charts	180
Other GDDM Features and Facilities	181
The Information Facility (TIF)	183
Table Definitions	183
Table Build, Update, Review	184
Prepare a Report or Chart from a Table	184
Structured Query Language/Data System (SQL/DS) for Programmers	185
The SQL/DS Preprocessor	185
The Use of SQL Commands	188
SQL/DS with Query Management Facility (QMF)	189
Advantages of Using QMF	190
A QMF Example in Two Query Languages	190

Application System (AS)

Overview

Application System (AS) is an integrated system for information handling, offering comprehensive facilities for data management and display. Its functions can be combined in many ways to satisfy the requirements of a wide range of applications:

- **Business Planning**
- **Statistics and Forecasting**
- **Project Control**
- **Document Preparation**
- **Business Graphics**

AS is a fully integrated system. You can use its tools individually or combine them to perform complex tasks. All applications in AS use common methods and conventions for data entry and storage, enabling a single set of data to be used for a variety of different purposes without the user needing to learn a new package.

Business Planning

With AS you can develop a model to plan ahead and try to predict how different assumptions will affect your business.

Business planning involves basic data-handling processes -- the creation of planning data, the calculation of results, the printing of reports, and so on. These needs are ideally met by a general information processing system such as AS. However, AS goes further by providing functions that are specific to planning, such as "what if" analysis and report consolidation.

Statistics and Forecasting

AS provides extensive facilities for statistical analysis and forecasting. All the analyses in AS statistics can be produced by giving a single command. In addition, certain analyses give a set of basic information and then allow you to derive further results interactively. To use AS statistics you need to tell AS two things -- what file to use and which analysis to perform.

The IN command specifies which file to use.

The STATISTICS command, used with keywords, tells AS what statistical or forecasting analysis you want and what data to use from the input file.

Applications for VM/SP

Project Control

AS can analyze arrow and precedence networks based on time or resource usage. Network drawings can be produced using four scales with multiple facilities for examining all or part of the network in selected detail. Resources can be assigned to tasks and analyzed to compare availability with usage, the results being displayed graphically. A risk analysis distribution of probable completion dates, either by activity or globally, can be displayed as a graph.

Document Preparation

The document preparation function of AS is comprised of two parts: MEMO and COMPOSE.

MEMO is the word processing facility. It fulfills the requirements for handling a small amount of text, enough for a normal business letter. It is screen-based and provides a way of immediately creating formatted documents. The document will be formatted whenever necessary and you will see the output on the screen as it would appear on paper. The formatting is done according to the rules you specify.

COMPOSE is the text processing facility of AS. It allows you to enter text on the terminal, with formatting statements, to produce documents in a suitable format. Its purpose is to format larger documents that require more formatting control than MEMO.

Business Graphics

AS offers a fully-interactive, syntax-free full-screen graphics interface. Data can be converted into line plots, scatter diagrams, surface charts, bar charts, 2 or 3-dimensional pie charts, radar charts, real or symbolic map drawings or combinations or charts on the same axes or displayed separately. High resolution graphics can be created with up to seven colors, and graphs can be combined with data, text and graphics from other files and areas of AS.

Reference

Application System General Information, GH45-5000, explains more about what AS is, what it can do, and what you need to install AS.

Application System Introduction User Guide, SC34-2209, introduces fundamental AS concepts and functions. It describes how to start using AS and the basic techniques for storing, manipulating, and displaying data. It includes examples.

Application System Business Planning User Guide, SH19-6301, introduces AS business planning facilities and the basic AS functions for building planning models. It describes how to add to and change plans, and design reports about them.

Application System Project Control User Guide , SH19-6305, describes how to do project analysis with AS. It tells how to use AS to define project networks, prepare data files and reports, do time and risk analysis, allocate resources, and display results.

Application System Document Preparation User Guide , SH19-6308, describes how AS is used to create, modify, format, display, and print documents.

Application System Statistics and Forecasting User Guide , SH19-6304, introduces the statistical techniques available on AS. Includes sections on how to prepare and analyze data with any of the statistical forecasting and testing techniques.

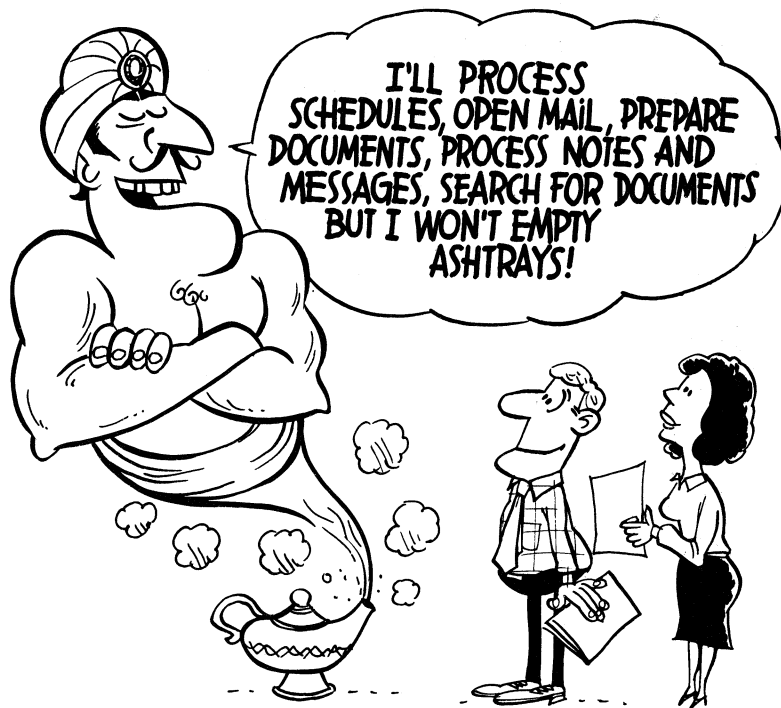
Application System Graphics Examples , SH19-6310, contains graphs and charts which can be produced in AS, each accompanied by the commands used to produce it.

Professional Office System (PROFS)

Overview

PROFS helps users do a variety of office tasks, including:

- Preparation of documents
- Distributing and receiving documents
- Formatting and printing documents
- Retrieving documents
- Handling routine office tasks.



Professional Office System (PROFS) is designed to help people who work in an office environment do their jobs more easily and efficiently. By automating many office tasks, PROFS promotes efficiency and reduces the adverse effects of human error and misplaced information. A lot of the clutter around the office disappears because much of the traditional paperwork is replaced by electronic correspondence. When documents are printed, use of predefined formats assures a consistent, more professional document quality.

With minimum training, executives, managers, secretaries, technical personnel, and clerical personnel can learn to use PROFS productively. Displayed prompting panels and program function key facilities help simplify its use.

Preparation of Documents

PROFS provides facilities that simplify the creation of documents such as letters and memos. Installations can design formatted prompts for information such as addressee, subject, reference, copy list, etc. An author profile facility allows information about the author, such as name, title, and department to be inserted automatically into a document. Text format control can be provided by the user as the text is entered, or provided automatically by the document preparation facility.

PROFS can also help users proofread completed documents. Each word is compared with entries in an internal dictionary. The system provides correct spelling for any misspelled or incorrectly keyed words. This facility can also locate words that are used incorrectly in the context of the sentence. It provides the user with alternatives for the misused words. In addition, the facility can locate and suggest alternatives to awkwardly used phrases.

Using PROFS, completed documents can be handled in a number of ways, including printing, filing, distributing, updating, and deleting them.

Distributing and Receiving Documents

PROFS allows users to send documents to other users on the system on which it is installed. When used with Remote Spooling Communications Subsystem Networking program product (RSCS), documents can be sent to users on other systems as well.

The person to whom a document is sent receives a summary about the document, containing information such as date, author, and subject. By reviewing this summary, the receiver can decide what action should be taken. For example, the receiver can print the document, view it on a display screen, file it, or pass it to another user.

Formatting and Printing Documents

Document layout can be controlled automatically by installation-defined format files, or by control information entered by the user at the time the document is created. Documents can be printed on VM/SP and RSCS-supported printers. An interface is provided for adapting printer output for printing on IBM 6670 and 6640 devices. Printing specifications, such as number of copies, can be prespecified for each user or selected by the user from a displayed menu.

Applications for VM/SP

Retrieving Documents

PROFS documents are indexed for easy retrievability using a variety of search arguments. Information such as who wrote the document, the date it was written, the name of the person to whom the document was sent, and the name of the subject is extracted for indexing. Other information, in the form of searchable keywords, can be included as search arguments by users.

Index records can be created for documents other than PROFS documents. This allows the PROFS search facility to be used in locating these documents.

Screens prompt users to fill in the necessary search arguments. These can specify author, date, etc. The user can display the results of a search at the terminal.

Handling Routine Office Tasks

In addition to document preparation and handling, PROFS can help users in doing administrative tasks. These include:

- Routing short, informal communications among users without the need of writing memos or letters
- Handling incoming electronic mail
- Creating, interrogating, and updating conference room schedules
- Creating, interrogating, and updating personal appointment schedules
- Scheduling follow-up action on documents
- Establishing an event reminder, which causes an audible tone at the terminal and displays specified information at a specified time and intervals.

Most of the displayed menus for using PROFS have associated help screens that can be accessed via program function keys.

Reference

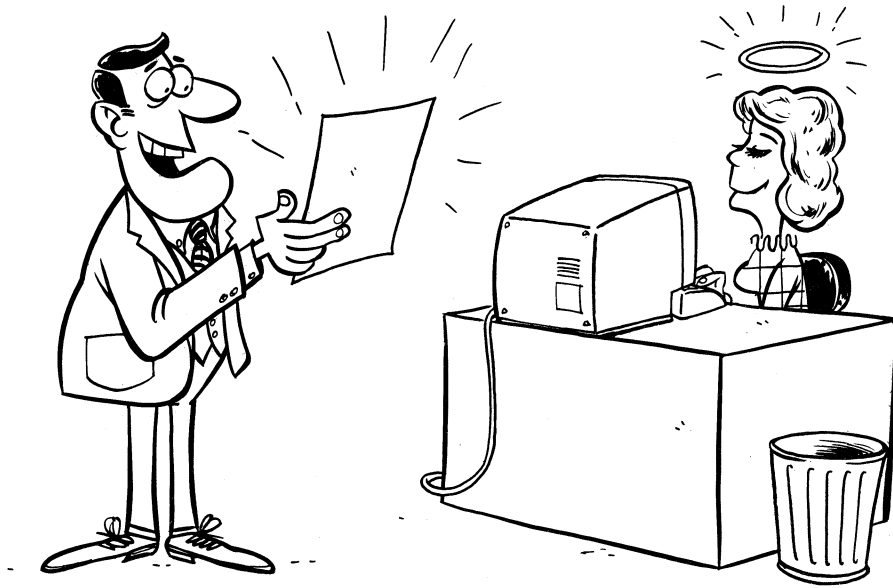
Planning for and Installing the Professional Office System, SH20-6800, describes planning, installing, preparation for use, and maintaining PROFS.

Using the Professional Office System, SH20-6797, describes how to use PROFS and provides related examples.

Document Composition Facility (DCF)

Overview

- DCF can be an invaluable tool for producing almost any kind of formatted text document, including such things as:
 - Specifications
 - Reports
 - Manuals
 - Manuscripts
 - Proposals.
- DCF provides document formatting capability through use of SCRIPT/VS control words and Generalized Markup Language (GML) tags.
- Once processed by the DCF formatter, a document can be printed, displayed, or used as input to other text handling programs.



Applications for VM/SP

With minimum training, people with no text formatting experience can learn to use DCF facilities to produce neat, professional looking documents that are easy to read and use. DCF provides clear advantages over other methods of document formatting, which include:

- Documents, even those produced by different people, tend to be consistent in appearance.
- Formatting standards are easier to manage, especially when a high-level formatting language such as GML is used.
- Document preparation can be much faster when irregular formatting is required. This includes such things as centered text, two-column text, variable size spaces for illustrations, etc.
- Document formats can be changed with minimum effort.
- A given text can be shared among several documents without the need of rekeying it for each one.

DCF provides a wide variety of text document formatting capability. When used in conjunction with the System Product Editor (described earlier in this book), almost any type of formatted document can be produced. DCF results in a consistency and accuracy that is very difficult to achieve when preparing a document manually, and it does it with far less effort.

Although its primary use is document formatting, DCF provides other functions. For example, it allows symbols embedded in documents to be used for such things as controlling conditional processing and entering characters that are not on the keyboard. It can be used for preparing documents for other text processing programs that require formatted documents as input.

SCRIPT/VS Is for Detailed Format Markup

One of the two major facilities of DCF is SCRIPT/VS. SCRIPT/VS provides:

- A set of formatting control words
- A document format processor.

SCRIPT control words are made up of two alphabetic characters, preceded by a period (.). The two characters usually indicate the function of the control word. For example, .fn is used for inserting a footnote into text. Control words are included in the text they format, usually at the point at which they affect the document's format.

SCRIPT/VS control words provide far more formatting functions than can be summarized in this discussion. However, here are a few to give you an idea of what control words do.

- .BX Enclose text in rectangular line.
- .CE Center text between left and right margins.
- .IM Embed another (specified) file into this file at this point.
- .IN Indent text a specified amount.
- .PA Eject the form on which the file is being printed to the top of the next page.
- .RF Print specified text at the bottom of each page of the printed document (running foot).
- .SK Skip a specified number of line spaces at this location in the document.
- .TC Insert an automatically generated table of contents at this location in the document.

SCRIPT/VS control words can be used for functions other than text formatting. For example, the .DU control word can be used to update the SCRIPT/VS dictionary of words. This dictionary is used by DCF for things such as checking the spelling of words in a file, if requested by a user.

GML Is High-Level Format Markup Language

The second major facility of DCF is Generalized Markup Language (GML). Using GML is a short cut to document formatting. GML provides a set of tags that define document elements rather than specific detailed format. A tag consists of a colon (:), followed by descriptive letters, and usually ending with a period (.). For example, the tag :p. starts a new paragraph. A tag expands into a sequence of SCRIPT/VS control words when it is processed. A programmer would think of a tag as a kind of macro statement.

Consider what goes into formatting lists of items, such as one in this paragraph. Using SCRIPT/VS control words for formatting, each detail of the list must be dealt with separately. This includes:

- Spaces before and after the list, as well as between list items
- Indents for the second and subsequent lines of each list item
- The bullet (●) character and spaces preceding each list item.

Using GML tags, formatting this list requires only:

- A :ul. tag to begin the list and a :eul. tag to end the list
- A :li.tag to begin each list item.

Another advantage to users is that format details need not be remembered. These include things like how many character spaces between the bullet character and the first character of the list item, or where to insert line spaces. Also, the sequence of control words (application processing

Applications for VM/SP

function, or APF) that get substituted for a tag can be modified as necessary to change the tag's effect on document formatting. This is done without changing the source document.

A starter set of GML tags is included with DCF. Users can expand this set by writing new APFs and assigning unique GML tags to them.

Processing DCF Formatted Documents

The DCF document processor program is invoked using the `SCRIPT` command. Many options of the `SCRIPT` command are available to users for controlling the processor. For example, the formatted output from the processor can be directed, by `SCRIPT` command option, to the user's terminal (default option), to a printer (`PRINT` option), or to a minidisk file (`FILE` option). Other options control such things as printing page numbers, checking of word spelling, selectively printing only certain pages, and suppressing messages.

Reference

DCF General Information, GH20-9158, provides an overview of the product.

DCF - GML Starter Set User's Guide SH20-9186, provides information about using the GML tags supplied with DCF.

DCF SCRIPT/VS Text Programmer's Guide, SH20-9161, provides more in-depth information about the product and its use.

DisplayWrite/370

Overview

- Full-screen interactive text editor/formatter supporting the IBM 3270 Information Display System and the IBM 3270-PC display terminal
- Optional support of the 3270 extended stream
- Context dependent help and tutorial
- User ability to set own defaults
- Command lists for special tasks
- Split-screen capability
- UNDO/REDO of system commands and local editing
- Basic and Advanced text editing capabilities
- Variable specification and merging support
- Support for pattern letters
- National Language Support including multi-language linguistic aids

Description

DisplayWrite/370 is a System/370 host text processing application package that includes functions for document creation and revision, interactive document formatting, printing, and linguistic support. DisplayWrite/370 is designed to be used without the need for extensive end-user training. This editor is both easy to learn and use for casual end users who are familiar with the working environment where DisplayWrite/370 is being used, but who are unfamiliar with text processing.

Applications for VM/SP

Screen Characteristics

Full-Screen Support

DisplayWrite/370 provides full-screen support for a minimum of 24 x 80 to a maximum of 62 x 160 screen sizes.

Extended Attributes

The editor provides optional support to the 3270 extended attributes featuring seven display colors, blinking, reverse video, and underscore options.

Document Representation

The screen will display the text in correct relative horizontal position only if the text is in 10/12/15 pitch uniformly. Mixed text in different pitches and/or proportional space fonts will not be displayed properly.

Cursor Movement

In edit mode, the cursor indicates where the editing operation should take place. Commands are issued through use of a PF key or command line. Use of a floating command line (displayed by pressing a PF key) maintains cursor position and decreases time spent using the command line.

Window Techniques

When the DisplayWrite/370 functions require interaction with the end user, the editor will overlay a small display area on the screen prompting the user to key a new value. By issuing the END or CANCEL command, the window disappears, and the original contents of the screen are re-displayed.

Operation

Because of the various formats available for entering commands, the DisplayWrite/370 accommodates both the infrequent and the experienced user. Entered commands are written to a log file for UNDO, REDO, and recovery purposes. An editor command can be issued in any one of the following ways:

- From menu windows
- From PF key
- From the command line
- From the floating command line
- From a command list

Error recovery

The user may revoke all changes made since the start of the editing session or since the last save command (whichever is more recent) through the UNDO command. The REDO command allows the user to invoke again any or all of the commands revoked by the UNDO command.

Help Facility

The HELP facility presents information on a specific subject including HELP on selected menus and messages using the window display technique. When HELP alone is requested, the HELP main menu panel is displayed, allowing the user to select the area in which assistance is needed.

Tailoring DisplayWrite/370

Most features of DisplayWrite/370 can be modified by the user and/or defined in profiles. These features include PF keys, Revisable Form Text Document Content Architecture (RFTDCA) controls and screen colors.

Split Screen

Split screen allows for the viewing of two different documents or two versions of the same document at the same time. The top half of the display may be edited while the bottom half may be viewed or copied.

Compatibility with Document Composition Facility (DCF)

DisplayWrite/370 provides an interactive formatting function, whereby the user may directly view the formatted document. DisplayWrite/370 does not support formatting functions that require two-pass processing; these may be handled by DCF.

Using DisplayWrite/370 under VM/SP, the user has the ability to convert most of the RFTDCA controls into SCRIPT controls.

Document Processing Options

- Create or revise an RFTDCA document.
- View any RFTDCA or Final Form Text Document Content Architecture (FFTDCA) document.
- Interactively paginate a document.
- Transform a RFTDCA document to a print data stream

Applications for VM/SP

Text Entry and Editing Capabilities

- Specify line and page format
- Insert and delete text
- Underline or emphasize text
- Block move, insert, copy and delete
- Tabs (left, right, center, decimal, comma, colon)
- Subscript and superscript (cannot be displayed)
- Overstrike
- Headers and footers
- Automatic line adjust (without Entry Assist)
- Auto hyphenation
- Global search/replace
- Find and change text (multi-directional)
- GoTo (page number, line number)
- Change font
- Instructions (insert controls)
- Defining and entering escape sequence
- Multiple formats (line, page, document, alternate)

Language Support

Linguistic Support

- Spelling verification
- Spelling correction assistance
- Automatic hyphenation
- Synonym support ⁸
- Grade level analyzer⁹

DisplayWrite/370 Language Dictionaries

The linguistic supports described above are available in the languages listed below.

- English Dictionaries¹⁰
- French and Canadian French
- German¹¹
- Italian
- Spanish
- Danish¹¹

⁸ UK/US English only

⁹ US English only

¹⁰ UK/US English, legal and medical

¹¹ Spelling correction assistance limited to non-compound words

National Language Support

National language support for all menus, HELP screens, and user messages is provided in the following languages:

Brazilian Portuguese	Canadian French	Danish
Dutch	UK/US English	French
German	Italian	Norwegian
Spanish	Swedish	

Extended character set support is provided through the customization functions performed by each customer during installation. This support includes both workstation character set and printer character set support.

Note: The IBM 3270-PC does not provide support for all listed languages at this time. Please consult your IBM marketing representative.

Invoking DisplayWrite/370

Under VM/SP and VM/SP HPO, DisplayWrite/370 is invoked from the CMS command line and will work with PROFS Version 2, Release 2.

Document Interchange

DisplayWrite/370 documents can be interchanged in either RFTDCA or FFTDCA by all products supporting the Document Content Architecture.

Reference

For more information refer to the following books.

Introducing DisplayWrite/370, GH12-5170

An information manual for general readers, including customer executives, data processing personnel, and professional end users. It contains short descriptions and example panels illustrating the functions provided by DisplayWrite/370.

Getting Started with DisplayWrite/370, SH12-5171

Using DisplayWrite/370, SH12-5172

DisplayWrite/370 Installation and Administration, SH12-5173

DisplayWrite/370 Diagnosis Guide, SH12-5175

DisplayWrite/370 Reference, SH12-5176

Using Graphics with DisplayWrite/370, SH12-5177

DisplayWrite/370 Quick Reference, SX12-5001

Graphical Data Display Manager (GDDM)

Overview

- GDDM can add general graphics capability to several IBM licensed programs.
- Some application areas include:
 - Adding graphics to business charts.
 - Interactive graphic programs that select or move displayed graphic representations.
 - Interface for displaying computer-designed or computer-aided drawings.
- Symbol editors help produce symbols such as company logos, scientific symbols, and fonts.
- GDDM can be used effectively for applications that use a mixture of graphic and alphanumeric data.

Because of its versatility, GDDM is used for graphic applications by several IBM licensed programs, both graphic and nongraphic. GDDM's primary task is to put text or graphic data into suitable form for display or printing. The graphics created may be business charts and graphs, engineering drawings, symbols, or even pictures defined by colored dots or lines.

GDDM Utilities for Drawing Charts

GDDM provides three interactive utilities that enable users to produce attractive charts quickly and efficiently.

The Interactive Chart Utility

This utility is a simple and flexible tool for drawing charts. Working interactively, a user can display a partially completed chart for inspection and making changes while it is being developed. For example, the format can be switched between line chart and bar chart. Some of the elements can be deleted and others emphasized. Detail and color can be modified, then the result viewed by simply pressing a program function key.

The Image Symbol Editor

This facility is typically used to add symbols such as company logos or special symbols to existing character sets. An image symbol is a rectangle of monochrome background on which colored dots define the symbol to be displayed or printed. For creating new image symbols, the Image Symbol Editor displays a matrix into which dots that define an enlarged version of the new symbol are keyed.

The Vector Symbol Editor

This facility is typically used to produce symbols that need to be displayed in various sizes. This is useful when size is used for emphasis. A vector symbol is defined by line strokes, not dots as are image symbols. When creating a new vector symbol on the matrix displayed by the Vector Symbol Editor, only the points that define the start and end of a line need be entered. Vector symbols can vary in size, and can be rotated or tilted. Compared to image symbols, they are more versatile, but display less accurately at small sizes.

Other GDDM Features and Facilities

GDDM provides an extensive set of two-dimensional graphic functions. These enable users to create pictures by describing them in terms of lines, arcs, colors, patterns, etc. The Interactive Map Definition feature of GDDM assists users in designing display screen formats. The names given to fields on the screen are the same names used in a program to refer to those fields. GDDM statements can be coded into application programs written in COBOL, FORTRAN, PL/I, Assembler, BASIC, and VS APL. *GDDM General Information* has more information about writing applications for GDDM. It also lists other IBM programs that can be of use to GDDM users.

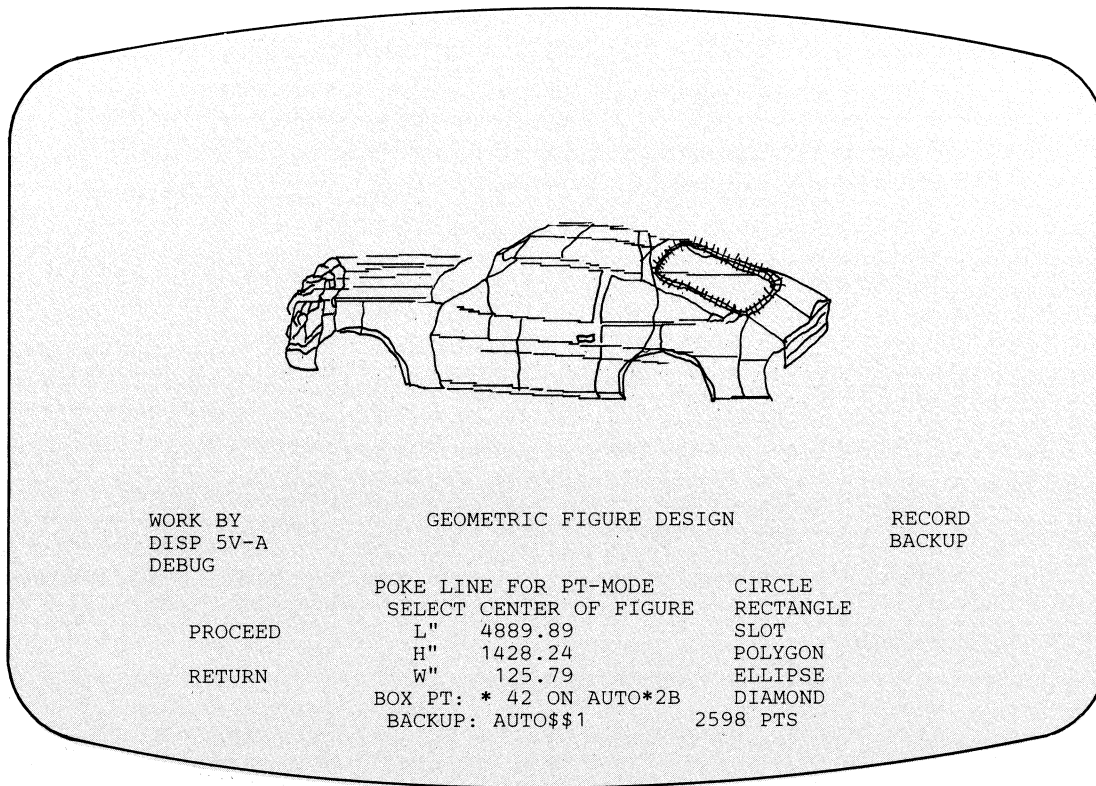


Figure 31. GDDM May Be Used for Displaying a Variety of Graphics

GDDM can be used with a variety of devices to produce both displayed and hardcopy graphics in monochrome or color. For example, using GDDM with the IBM 3279 Video Output Feature and a suitable projector is an effective means of producing high-quality visual aids for presentations. Graphics can be printed in four colors on the IBM 3287 Printer. A Print Utility function provides for printing both graphic and alphanumeric output.

GDDM can be used in batch processing mode. This allows routine production programs or time-consuming jobs to be run during times when the processing computer load is light.

Reference

GDDM General Information, GC33-0100, presents a general description of GDDM, its use, and some IBM programs that use it.

The Information Facility (TIF)

Overview

The Information Facility (TIF) is an interactive program that enables users to manage information and create applications.

Some of the routine tasks that users can do with TIF:

- storing their own data or accessing existing data
- reviewing, updating, deleting, and adding to the data
- producing reports or charts from the data.

TIF's options are based on the idea of tables. A table is an arrangement of data in columns and rows. TIF allows users to:

- define tables
- build, update, and review tables
- prepare and define reports and charts.

TIF also has "secondary" functions:

- Dictionary functions - displays names of tables, screens, reports and charts that the user has defined.
- Table functions - move, reformat, rename, delete, or sort tables. The user can also extract selected rows from a table.
- Conversational editor - prompts allowing the user to edit table definitions, screens, reports, and charts.

Although all data used appears to the user as TIF data, it can be a CMS file or a relational table managed by SQL/DS.

Table Definitions

Table definitions are the essential facts about a table:

- its name
- maximum number of rows needed
- column information:
 - names
 - widths

Applications for VM/SP

- data types
- which ones are keys

Table Build, Update, Review

TIF displays a prompt screen on which you specify a table and the names of the columns you want to display. In specifying the table and columns, and giving some other optional information, you are, in TIF's terms, defining a screen. When you run the screen, TIF shows you the table (empty if it is new) and the columns, and you build, update, or review on the terminal screen.

Build means the initial entry of rows into the table.

Update means adding or deleting rows or changing the data in the columns.

Review means displaying the table on the screen.

Prepare a Report or Chart from a Table

TIF allows you to view the data in a table in various chart forms (bar, pie, line). To define the report you select this "Prepare" option.

Reference

Introduction to The Information Facility, GC26-4217, explains how to start TIF, prepare simple reports, charts, and graphs, update and review data.

The Information Facility: Concepts and Functions, SC26-4204, explains the concepts and functions of TIF and provides a methodical grounding in what TIF can do. This book is for users who want to understand TIF thoroughly and prepare themselves to become developers of TIF applications.

Structured Query Language/Data System (SQL/DS) for Programmers

Overview

- **SQL commands can be embedded in application programs written in:**
 - Assembler
 - COBOL
 - PL/I
 - FORTRAN.
- **Embedded SQL commands are analyzed and converted into a module that is accessed for SQL/DS functions when the host program is executed.**

Data in SQL/DS data bases can be accessed by programs running in CMS virtual machines as well as accessed interactively from a terminal. SQL/DS supports programs written in Assembler, COBOL, PL/I, or FORTRAN. Figure 32 on page 187 shows examples of SQL commands embedded into programs of each of the supported languages. Using the power of embedded SQL commands reduces the programming needed to provide data handling capability.

The SQL/DS Preprocessor

Preprocessors perform two functions:

1. They cause the SQL commands to become comment statements in the source code, and add statements in the host program's language to invoke communication routines to replace the SQL commands. The changed version of the code is used as input for compiling (or assembling) the host program.
2. They convert the SQL commands into a module (access module), containing the machine code equivalent of the SQL commands. They then store the access module in the SQL/DS data base.

Any embedded SQL commands must be analyzed and converted to an access module by the SQL/DS preprocessor facility, PREP, before the host program is compiled (or assembled). When the application program is executed, it calls the access module to handle the SQL commands.

Applications for VM/SP

Using SQL/DS Extended Dynamic Statements, installations can write their own preprocessors for other host programming languages.

COBOL Example:

```

DATA DIVISION.
WORKING-STORAGE SECTION.
    EXEC SQL BEGIN DECLARE SECTION END-EXEC.
01  XX.
    49 XX-LENGTH PICTURE S9(4) COMPUTATIONAL.
    49 XX-VALUE  PICTURE X(24).
77  YY PICTURE S9(9) COMPUTATIONAL.
77  ZZ PICTURE S9(4) COMPUTATIONAL.
    EXEC SQL END DECLARE SECTION END-EXEC.
    •
    •
PROCEDURE DIVISION.
    EXEC SQL SELECT DESCRIPTION, QUANTITY INTO :XX, :YY
    FROM INVENTORY WHERE PARTNUMBER = :ZZ END-EXEC.

```

PL/I Example:

```

EXEC SQL BEGIN DECLARE SECTION;
    DCL XX CHAR(24) VAR;
    DCL YY BIN FIXED(31);
    DCL ZZ BIN FIXED(15);
EXEC SQL END DECLARE SECTION;
    •
    •
EXEC SQL SELECT DESCRIPTION, QUANTITY INTO :XX, :YY
FROM INVENTORY WHERE PARTNUMBER = :ZZ;

```

Assembler Language Example:

```

    EXEC SQL BEGIN DECLARE SECTION
XX      DS      H,CL24
YY      DS      F
ZZ      DS      H
    EXEC SQL END DECLARE SECTION
    •
    •
    EXEC SQL SELECT DESCRIPTION, QUANTITY INTO :XX, :YY      C
    FROM INVENTORY WHERE PARTNUMBER = :ZZ

```

FORTRAN Example:

```

    EXEC SQL BEGIN DECLARE SECTION
        CHARACTER*24  XX
        INTEGER       YY, ZZ
    •
    EXEC SQL END DECLARE SECTION
    •
    EXEC SQL DECLARE C1 CURSOR FOR
*      SELECT DESCRIPTION, QUANTITY
*      FROM INVENTORY
*      WHERE PARTNUMBER = :ZZ
    EXEC SQL OPEN C1
    EXEC SQL FETCH C1 INTO :XX, :YY
    EXEC SQL CLOSE C1

```

Figure 32. Examples of SQL Commands Embedded in Programs

Applications for VM/SP

The Use of SQL Commands

Any SQL commands used for doing operations from a terminal can be used as embedded statements in an application program. However, there are some SQL commands designed to be used only as embedded statements in an application program.

SQL/DS provides facilities (PREPARE and EXECUTE) for handling SQL commands issued dynamically while an application program is executing. For example, issuing queries and receiving results interactively while running an application. The special facilities are needed because the SQL/DS preprocessor does not process these SQL commands as embedded program statements.

Some special rules apply for coding SQL commands into an application program. For example, program variables to be shared by SQL/DS and the host program are declared normally, but the section of program in which they are declared is preceded and followed by special commands. Also, program variables are preceded by a colon (:) when referred to from embedded SQL/DS statements.

Reference

SQL/Data System Concepts and Facilities for VM/SP, GH24-5065, provides an overview of the product's facilities related to programming.

SQL/Data System Application Programming for VM/SP, SH24-5068, describes the product from the application programmer's point of view.

SQL/DS with Query Management Facility (QMF)

Overview

- **QMF (Query Management Facility) provides an easy-to-use, interactive means of accessing and manipulating data in a SQL/DS data base.**
- **Key features of QMF are:**
 - **Full-screen interaction with users**
 - **On-line help facility**
 - **Choice of two query languages**
 - **Use of simple and familiar data model of tables**
 - **Interactive tailoring of reports**
 - **Preparation of data for graphics presentation.**

Structured Query Language/Data System (SQL/DS) is a data base management system for manipulating data stored in row and column format (tables). A basic description of the product is in Chapter 5.

This topic discusses the combined facilities of SQL/DS and Query Management Facility (QMF). QMF is a program product that is designed to make the use of SQL/DS considerably easier and more effective. It serves as a communications interface between users and SQL/DS. QMF runs in a CMS virtual machine, together with SQL/DS and any other program products required to take advantage of its full capabilities.

In addition to retrieving data and displaying it on a screen, QMF allows users to:

- **Format retrieved data into a report**
- **Include totals, subtotals, averages, maximums, and minimums in a report**
- **Update, insert, delete information in data tables**
- **Add new tables to a data base**
- **Define various views of data from one or more tables.**

Applications for VM/SP

Advantages of Using QMF

QMF is easy to learn and to use. It requires no prior data processing experience. The few commands that must be learned have simple, easy-to-remember names like DISPLAY, RUN, PRINT, SAVE, HELP, and END. Users can focus their attention on the job to be done rather than how to do it.

There is interactive communication between QMF and users. Prompts are displayed to assist users in providing input. Full-screen communication facilities greatly simplify the process of providing information to the system.

Program function keys can be used to initiate many functions. On-line help is available to users if needed. In case of a user error, informative messages help get the user back on track. Users can request help on operations or error messages by pressing a program function key.

QMF accepts input in either of two query languages:

- SQL provides a simple English-like language. It requires the user to learn only a few basic rules, such as keyword placement.
- Query By Example (QBE) displays a graphic representation of data tables on which users enter an “example” of the output they want to see.

A QMF Example in Two Query Languages

This example, illustrated in Figure 33 on page 191, is not intended to be used as a guide or reference for using QMF. The purpose of the example is to point out:

- The ease with which SQL or QBE language can be used with the QMF facility
- Use of the full-screen facility for communicating with QMF
- One type of data presentation that can be requested
- Similar results from using either query language.

The top screen in Figure 33 on page 191 shows how an SQL query might be entered to request specified information from a data table named STAFF. The entire query, shown in color, must be entered.

The middle screen in Figure 33 on page 191 shows how a QBE query might be entered to request the same information. The grid, displaying the name of the table and the columns containing the information requested and the names of the columns, resulted from a prior DRAW STAFF command. Only the characters shown in color must be entered.

The bottom part of the figure shows the results of either the SQL or QBE query. The user can get a printed copy of the results by pressing the PRINT program function key (4).

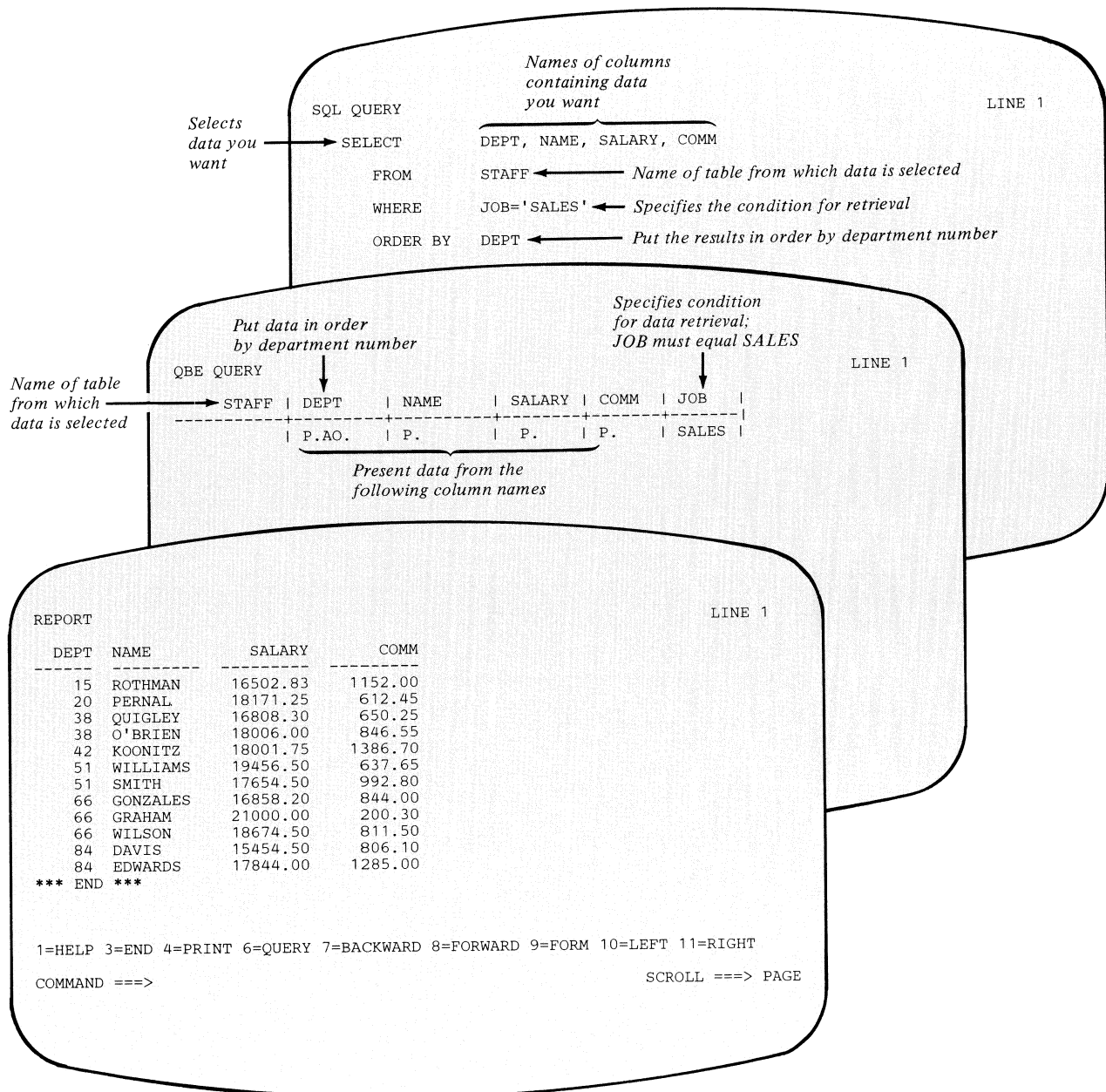


Figure 33. SQL/DS Responds to Either SQL or QBE Queries

Applications for VM/SP

Reference

SQL/DS Concepts and Facilities for VM/SP, GH24-5065, describes SQL/DS for administrators, application designers and programmers, and system programmers.

Introduction to Query Management Facility, GC26-4101, describes the product. together with installation tasks and requirements.

Chapter 7. Programming Language Products

This chapter discusses programming languages that provide additional application capability for VM/SP users.

The products described in this section are intended for anyone involved in planning or writing application programs. Most programming language products are designed to be used by application programmers. Some, such as Assembler, are of special interest to system programmers.

Additional sources of information about programs that run on VM/SP are:

- *Software Directory*, GB21-9949
- *The Software Catalog*, G320-6531

Chapter Contents

OS/VS COBOL	195
The Language and the Compiler	196
Program Development Aids	196
Advanced Program Applications	197
The OS/VS COBOL Subroutine Library	197
PL/I	199
The Language and the Compiler	199
Compile-Time and Execution-Time Facilities	200
Potential Benefits of Using PL/I	200
Subroutine Libraries Provide Many Useful Functions	201
Assembler Language	203
Assembler Fills Specific Programming Needs	204
The Assembler Program	204
VS FORTRAN	206
FORTTRAN Interactive Debug	209
Functions Provided	209
Some Examples of How Interactive Debug Works	210
VS APL	211
Potential Benefits	211
Features and Facilities	212
IBM BASIC	214
Display Management System for CMS (DMS/CMS)	217
Panel Creation under DMS/CMS	218
Application Programs and Procedures for Using Panels	218

OS/VS COBOL

Overview

- COBOL is a programming language for use in commercial applications.
- The OS/VS COBOL compiler offers:
 - Program development aids
 - Advanced program applications
 - Efficient object time performance
 - Productive compile time performance.
- The OS/VS COBOL Library Management Facility optionally allows a single copy of the COBOL library subroutines to be shared by all currently executing COBOL programs.



Programming Language Products

The Language and the Compiler

COBOL (COmmon Business-Oriented Language) is a high-level programming language that is widely used for commercial data processing applications. It is a powerful language, yet its English-like format makes it easy to learn and easy to use. OS/VS COBOL provides advanced programming facilities designed to reduce development time, resulting in increased programmer productivity.

The OS/VS COBOL program product consists of a source language compiler and a subroutine library. The subroutine library is available as a separately orderable program product.

Program Development Aids

Eased programming rules simplify program development. For example, relaxed punctuation rules allow a space before a period, semicolon, or colon.

Debugging features allow the user to specify conditions under which data items or procedures are to be monitored during program execution.

A Federal Information Processing Standard (FIPS) flagger identifies source clauses and statements that do not conform to a specified FIPS level. A default level can be established at installation time, and overridden at compile time if desired.

The compiler can provide output directly to the user's terminal. This output might include such things as compilation progress messages, diagnostic messages, or the program listing.

A Lister facility permits the user to get a reformatted, detailed source code listing. This listing contains complete cross-reference information at the source level. This facility also produces a summary listing that contains selected statements from the source program, plus a condensation of the detailed information from the reformatted source listing.

Other options are available that identify all verbs, and provide statistics about the most heavily used portions of the source program.

A debugging facility provides diagnostic information in COBOL format instead of a hexadecimal format. No source language changes are required. The debugging information is requested through object time control statements. Thus, multiple debugging runs can be made without recompilation.

A flow trace option allows users to receive a formatted trace for procedures executed prior to an abnormal termination. The number of procedures to be traced is specified by the user. This option requires no source language changes.

Users can request a conditional or unconditional syntax-checking compilation. The unconditional option scans the source for syntax errors,

but does not generate object text. The conditional option produces a full compilation if nothing worse than a minor error occurs. If a more serious error occurs, no object code is produced.

A statement number option provides information about the statement being executed at the time of an abnormal termination. The program containing the statement and the statement number are identified, and if there is more than one verb in the source statement, the verb being executed is identified. This option requires no source language changes.

Advanced Program Applications

OS/VS COBOL programming advancements available to the programmer include:

- Enhanced VSAM support
- Expanded physical sequential file capability
- Added communication support
- Expanded library facilities
- Powerful data manipulation
- Extended computational facilities
- User-defined collating sequences.

The OS/VS COBOL Subroutine Library

The OS/VS COBOL subroutine library is a partitioned data set residing on a direct access storage device. It contains COBOL object-time library subroutines in load module form. This library is a separately orderable program product. One copy is shipped with the OS/VS COBOL program product.

The library subroutines perform execution-time operations that require either repetitive or extensive code. This can significantly reduce the size of the program object module. Library routines needed to execute the problem program are either combined with the object module at link-edit time, or they are dynamically fetched during program execution. Some of the types of library subroutines are:

- Input/Output (excluding VSAM)
- Conversion
- Arithmetic verbs
- Other verbs
- Sort/Merge interfaces

Programming Language Products

- Checkpoint/Restart
- Segmentation feature
- Communications
- Debugging
- VSAM.

Reference

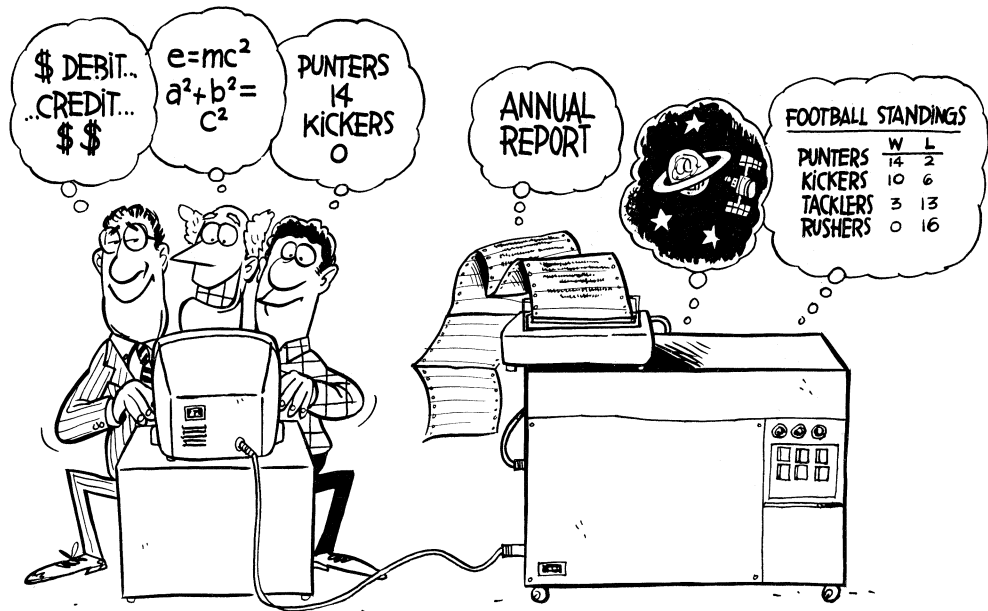
OS/VS COBOL General Information, GC28-6470, contains information to aid systems planners and analysts in evaluating the product. This information includes performance, compatibility, and system requirements statements, and lists restrictions for using OS/VS COBOL under CMS.

CMS COBOL User's Guide, SC28-6469, provides information about how to use COBOL under CMS.

PL/I

Overview

- PL/I is a powerful general-purpose programming language.
- The OS PL/I Optimizing Compiler offers both extensive optimizing and extensive debugging aids.
- Subroutine libraries are available, both as part of a composite compiler/library program product and as separate program products.



The Language and the Compiler

PL/I (Programming Language/One) is a powerful, comprehensive language that can be used to solve both business and scientific problems. The OS PL/I Optimizing Compiler (PL/I compiler) has incorporated into it the best features of several other proven compilers. It is intended to meet the needs of PL/I users in medium to large-sized installations.

PL/I statements are written in a high-level, easy-to-learn format, much like English or elementary algebra. The language's primary users are commercial programmers, scientific programmers, system programmers, and any users who need a personal computing language.

Programming Language Products

A PL/I program consists of one or more blocks of statements called procedures or subroutines. One PL/I procedure can invoke another PL/I procedure.

Compile-Time and Execution-Time Facilities

A wide variety of compiler options are available to the PL/I programmer for specifying information or requesting optional compiler facilities. The options allow the programmer to do such things as manage the compiling environment, control syntax checking, control output from the compile process, and manage diagnostics. All options are outlined in *PL/I Optimizing Compiler: General Information*.

Program compilation can be performed either interactively or through use of batch processing. Processor control statements that affect the text of the source can be inserted in a PL/I program. For example, the %INCLUDE statement might be used to copy an installation's standard set of definitions into a program.

PL/I programs can:

- Invoke Assembler, COBOL, or FORTRAN programs
- Be invoked by COBOL or FORTRAN programs
- Access data sets created by COBOL or FORTRAN programs.

PL/I programs can communicate with programs in other languages if an Assembler language interface is provided.

PL/I program statements can invoke other program products to request special services. This means that records can be passed to a program, sorted, and then returned to the PL/I program.

Potential Benefits of Using PL/I

PL/I is a problem oriented rather than a machine oriented language. This gives the advantages of:

- Less hardware knowledge required
- Programmers can concentrate on what is to be done rather than how to do it
- A minimum of machine and device dependence
- Easier hardware conversion.

Using PL/I results in faster overall programming, debugging, and program maintenance. A variety of aids make program debugging more effective. Program maintenance is made easier through:

- Better code readability

- Testing facilities that may remain in the program
- Identification of “dead” code.

An important feature of PL/I is the assignment by the compiler of default values for unspecified name attributes or statement options. This reduces the amount of declaration and other program writing required. It also makes it possible to learn a subset of the language without knowing all the possible alternatives.

Subroutine Libraries Provide Many Useful Functions

Two subroutine libraries are available for use with OS PL/I. One library consists of resident subroutines, and the other consists of transient routines. The compiler and the libraries can be ordered separately or as a package.

Functions provided by the resident library subroutines are:

- Mathematical subroutines
- Data type conversions
- Edit-, list-, and data-directed I/O (stream-oriented transmission)
- Program initialization
- Storage management
- Display
- Timer facilities
- Error handling.

Functions provided by the transient library subroutines are:

- Print and message subroutines of the error and interrupt handling subroutines
- File opening and closing
- Record-oriented and stream-oriented transmission.

Programming Language Products

Reference

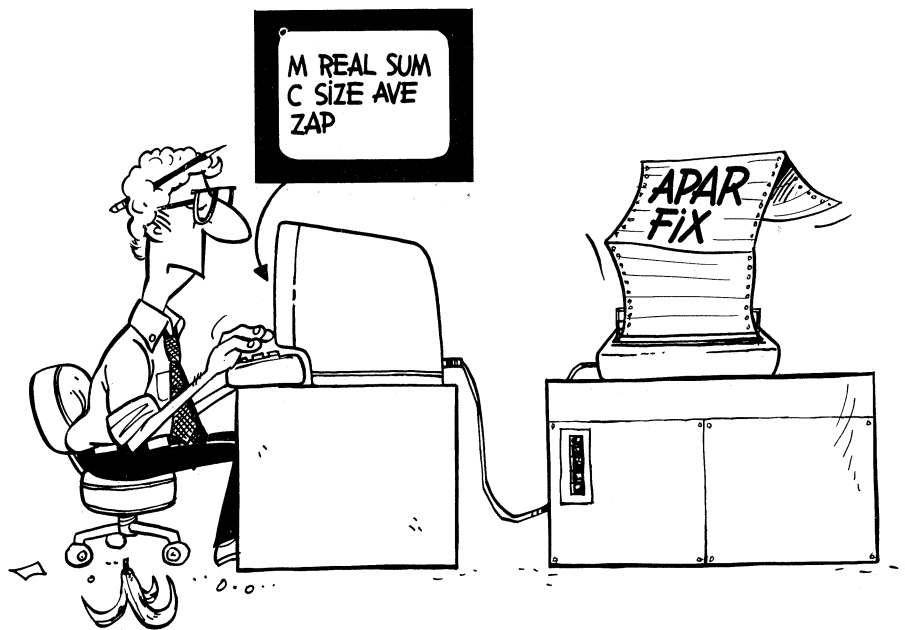
OS PL/I Optimizing Compiler: General Information, GC33-0001, introduces installation managers, analysts, and programmers to the facilities available with the OS PL/I Optimizing Compiler.

OS PL/I Optimizing Compiler: CMS User's Guide, SC33-0037, tells programmers how to use the OS PL/I Optimizing Compiler under CMS to compile, debug, and run PL/I programs.

Assembler Language

Overview

- Assembler is a symbolic language that closely maps (one-to-one) to the machine language instructions that result from it.
- The primary users of Assembler language are system programmers.
- An Assembler language processing program is shipped with the VM/SP system.
- Assembler H is a program product that can be ordered separately.



All programming languages, other than Assembler, discussed in this book are high-level languages. Normally, each statement of a program written in one of these languages results in several machine language instructions being generated when the source program is *compiled*. Using high-level languages generally makes writing programs easier and faster.

On the other hand, each source statement of an Assembler language program normally results in one machine language instruction being generated when the source program is *assembled*. Writing programs in Assembler language is usually more time-consuming, and is generally considered more difficult than using high-level languages.

Programming Language Products

Assembler Fills Specific Programming Needs

Assembler language gives programmers a degree of program control that is not available to them when using the high-level languages. There are times when coding in Assembler is a requirement for a particular application. System programmers use Assembler language for doing such things as:

- Maintaining programs written in Assembler language
- Coding functions at a byte or bit level, not provided for by a high-level language
- Writing routines, such as special interfaces, that must be coded in Assembler language.

Macro instructions are available to Assembler language programmers. A macro instruction is a program statement that results in a sequence of machine language instructions being generated, in much the same way a statement of a high-level programming language expands when it is compiled. Macro routines (definitions) provide a fast and easy way to program functions such as input/output and data management operations. Users can code their own macro definitions and invoke them from Assembler programs.

The Assembler Program

The latest version of the Assembler H program product can be ordered separately. This assembler runs under control of CMS. With minor exceptions, it is compatible with earlier assemblers. It will accept any error-free programs that have correctly processed on earlier level assemblers. System programmers can change the assembler to tailor it to their installation's needs.

The assembler is invoked under CMS using the HASM command. Default values are provided for all options of the HASM command, but alternative values can be specified. The option default values can be set to suit the needs of each installation.

The input to the assembler is a program written in Assembler language. The filetype of the source program file must be ASSEMBLE. This input is called a source module. The output of the assembler consists of a program listing (LISTING file) and an object module (TEXT file).

The assembler has many diagnostic features to aid in the location and analysis of program errors. These include:

- Diagnostic messages for various stages of the assembly process
- A macro trace facility
- A specially formatted dump when an assembly cannot be completed.

Reference

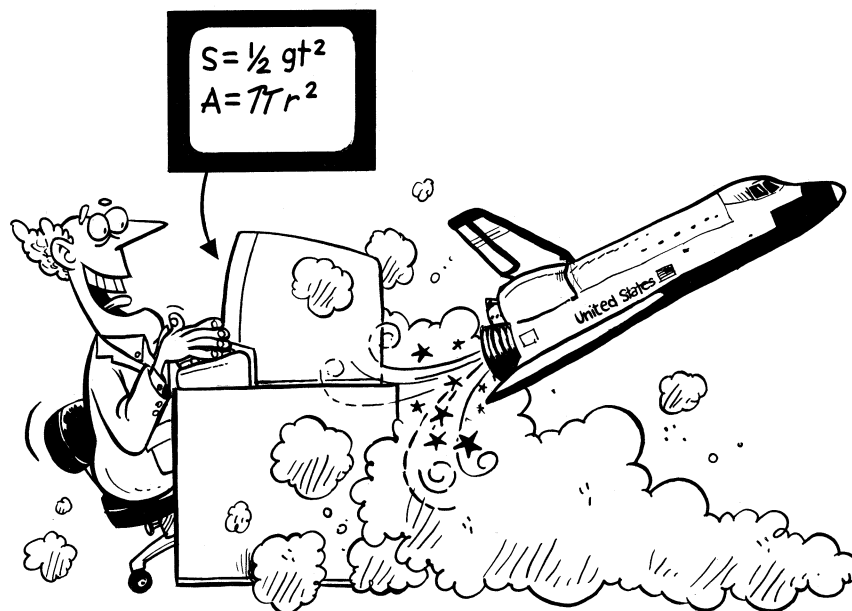
Assembler H Version 2: General Information, GC26-4035, describes the Assembler program, with emphasis on the most recent enhancements.

Assembler H Version 2: Application Programming Guide, SC26-4036, describes how to use the assembler program.

VS FORTRAN

Overview

- FORTRAN programming languages are used primarily for scientific and engineering applications.
- FORTRAN program statements look and read much like mathematical equations.
- The VS FORTRAN program product has added features that result in more effective and efficient FORTRAN programming.



FORTTRAN is derived from FORMula TRANslation. It is a mathematically oriented high-level programming language. Applications range from simple problem solving to large scale numerical systems that use either simulation or optimization techniques.

The VS FORTRAN compiler and an execution-time library are shipped together as a program product. The library, which is separately orderable, includes:

- Mathematical, character, and bit functions
- External data conversion routines
- Input/output routines

- Service routines
- Error routines.

Two levels of FORTRAN programs can be processed by the VS FORTRAN compiler:

- FORTRAN 77 standard, plus IBM extensions.
- FORTRAN 66 standard, plus IBM extensions.

The level is specified to the language processor at compile time or a default value can be used.

Some features of VS FORTRAN that make it excellent for application program development are:

- The VS FORTRAN compiler program and library modules are reentrant. Because of this, they can be made resident in virtual storage, with one copy serving the needs of multiple concurrent compilations.
- VS FORTRAN programs can be compiled on one of the supported VS operating systems, then link edited and executed on any other supported VS operating system.
- Installation of VS FORTRAN is done in two easy steps:
 - Set up and define the VS FORTRAN libraries.
 - Code and execute the installation control statements.
- VS FORTRAN makes difficult application programs easier to develop than in the past:
 - Easy to use VSAM support.
 - Improved I/O language that gives users more control when processing FORTRAN files, and more information about each record processed.
 - Using internal files, users can move data from one internal area to another, while converting it from one format to another.
 - Facilities provide for top-down program design and for structured programming.
 - The CHARACTER data type gives users more flexible control over specification and processing of nonnumeric data.
 - Additional built-in (intrinsic) functions.
 - Constants can be defined with symbolic names.

Programming Language Products

- VS FORTRAN aids for program development include:
 - Programs can be written in free format, which avoids many of the coding restrictions associated with writing code in fixed format.
 - Flags are provided to help users in creating programs that conform to the specified standard language level.
 - Debugging “packets” can be specified at the beginning of a source program to give users execution time diagnostic information that is easy to understand and use.
 - Diagnostic information issued by the compiler and by the execution time library is more precise and more informative than in the past.

Reference

VS FORTRAN Compiler and Library: General Information, GC26-3983, provides an overview of the product. It includes compatibility and requirements information.

VS FORTRAN Application Programming: Guide, SC26-3985, gives guidance information about designing, coding, compiling, executing, and debugging VS FORTRAN programs.

VS FORTRAN Application Programming: Language Reference, SC26-3986, gives rules for coding source programs.

FORTRAN Interactive Debug

Overview

- **FORTRAN Interactive Debug** allows the debugging of FORTRAN programs interactively from the user's terminal.
- **Interactive Debug** subcommands enable programmers to control, monitor, and change executing FORTRAN programs.



FORTRAN Interactive Debug provides users with the ability to debug FORTRAN programs conversationally. It is a program product that runs under CMS. Use of Interactive Debug can result in:

- Simplified interaction with executing FORTRAN programs
- Increased productivity through simpler debugging procedures
- More reliable code.

Functions Provided

Interactive Debug allows users to:

- Start and stop an executing program
- Examine and change the value of variables
- Display all or parts of a source program

Programming Language Products

- Trace program transfers
- Request help
- Track frequency of execution
- Respond to notification of errors.

Some Examples of How Interactive Debug Works

Generation of Interactive Debug information is specified by the TEST compile-time option. Debugging is initiated by issuing the TESTFORT command with suitable options. Thereafter, the user is prompted each time the system is ready to accept a subcommand.

Like other CMS subcommands, Interactive Debug subcommands are simple and easy to use. For example, to tell the system to stop execution at program line number 50, enter:

```
AT 50
```

To tell the program to begin execution, simply enter:

```
GO
```

To request the system to display the number of the program statement currently executing, enter:

```
WHERE
```

Transfer of control (branch) within a program can be monitored by issuing the TRACE subcommand. It tells the user for each branch:

- The program statement that caused the branch and the statement that received control
- Each time a subprogram was entered, and each time control returned to the main program.

These are only a few of the Interactive Debug subcommands available, but they illustrate the kinds of functions provided and the ease with which they can be used.

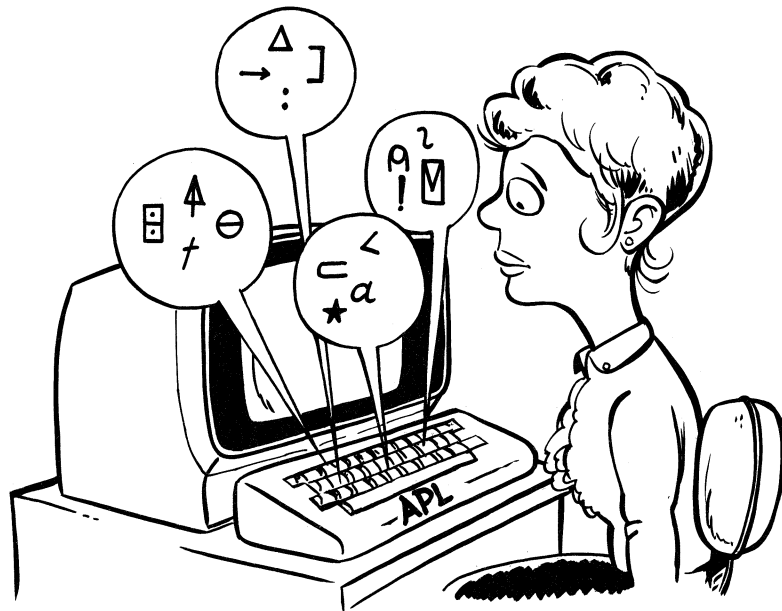
Reference

FORTRAN Interactive Debug for CMS: Guide and Reference, SC28-6885, describes FORTRAN Interactive Debug commands, tools, procedures, and messages.

VS APL

Overview

- The VS APL program product interprets the APL programming language and executes system commands and system functions to control the APL environment.
- VS APL can be used interactively or in batch processing mode.



APL is an easily learned and easily used programming language that can be used for a variety of problem-solving applications. Its users include financial analysts, engineers, students, mathematicians, and application programmers. VS APL is a powerful facility for using the language for both business and scientific applications. It can be used interactively from any one of the many device types it supports as terminals. VS APL can also operate as a batch processor under CMS, using the CMS batch facility.

Potential Benefits

There are many potential benefits to VS APL users. They include:

- Faster problem solving, data analysis, interactive program development, and interactive production computing.
- Easy-to-use APL applications are available.
- Improved security through access control, interpretive mode of operation, and locked functions.

Programming Language Products

- Time and cost savings through new personal computing and end-user applications.
- Interactive data analysis facilities to complement a data base system.
- Flexibility in tailoring applications to suit user's individual needs.

Features and Facilities

VS APL allows users to communicate with auxiliary processors. An auxiliary processor is a program that communicates with an APL program through shared variables. This permits APL users to request special host-dependent tasks, such as external file input or output operations. Several auxiliary processors that can be used under CMS are shipped with the product. Examples of the kinds of functions they provide include:

- Allow APL users to issue some CP and CMS commands without leaving APL mode.
- Provide access to CMS disk files.
- Allow APL users to add entries to the CMS console stack.
- Interact with the Graphical Data Display Manager (GDDM) program product to control certain user display or printer terminals.

System programmers can create others to satisfy the needs of an installation.

VS APL provides a session manager, which produces a continuous record of the VS APL session events. Session manager commands enable users to display and set several parameters that affect VS APL sessions, such as the size of the display screen, values of program function keys, and the size of the session log. Use of the VS APL session manager requires an IBM 3270 terminal and the Graphical Data Display Manager (GDDM) program product.

Conversion aids supplied with VS APL help users migrate from prior implementations of APL. These same conversion programs can be used to transfer functions and data from users of other versions of APL. The VS APL conversion program gives an installation the option of doing only format conversion, or both format and content conversion. Format conversion refers to changes needed in the internal format of directories and workspaces. Content conversion refers to changes required to allow for the effects of APL languages differences within defined functions.

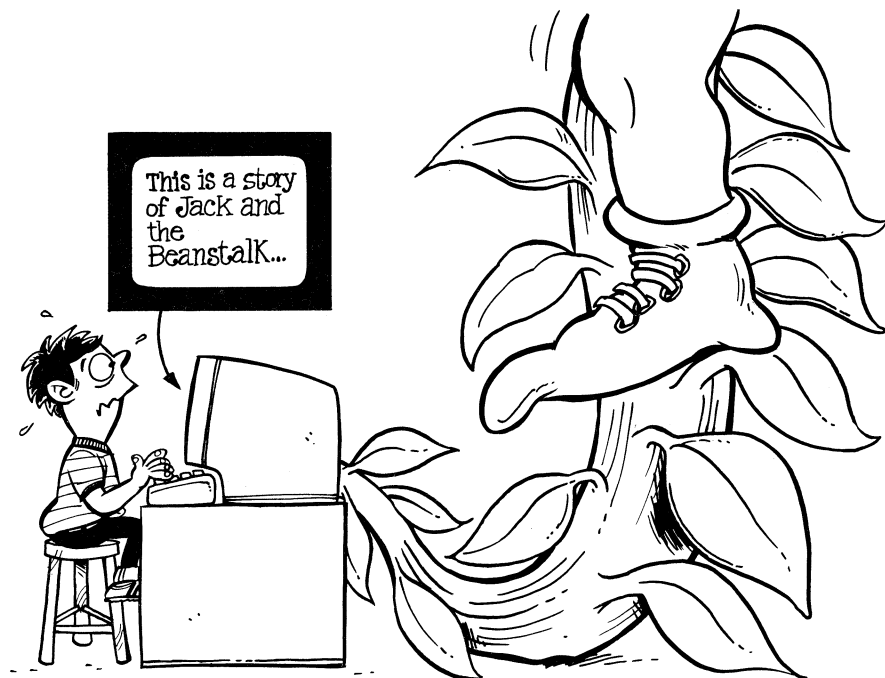
Reference

VS APL General Information, GH20-9064, provides an introductory description of VS APL for data processing managers, system designers, and application designers.

VS APL for CMS: Terminal User's Guide, SH20-9067, describes how to use the product.

IBM BASIC

- The IBM BASIC environment provides users with all the tools needed for easy management of programming activities. The environment shields users from the underlying operating system structure.
- Users can get a response to selected IBM BASIC statements as soon as they enter the statements into the system.
- Running IBM BASIC in batch mode, users can compile and combine main and subprograms together, or they can combine these programs dynamically at run time.
- The processor and the library routines are reentrant; many programs can share one copy maintained in shared virtual storage.



BASIC was developed to be an easily learned general problem-solving language. IBM BASIC is a new-generation of the BASIC programming language. It provides users with many powerful development capabilities. Some are reflected in the improved BASIC language, and some by the environment surrounding the language.

While remaining simple and easy to use, IBM BASIC adds power and flexibility to the language. It provides a complete programming environment, including its own language-sensitive command set. It allows users to intermix BASIC source statements with editing, debugging, and library management commands. IBM BASIC statements can be grouped into a program or, in some cases, executed immediately just like commands. The IBM BASIC environment is complete and self-contained, so users are generally free of operating system concerns.

IBM BASIC provides extensive capabilities for writing source programs using English-like statements. It can be used for a broad range of applications. Its purpose is not limited to a specific application environment, such as scientific, or business. Some of the features that make IBM BASIC a good general-purpose programming language are:

- Statements, such as the DO loop and IF/THEN/ELSE, facilitate structured programming design and development.
- Variables and statement labels can be given meaningful names of up to 40 characters.
- Versatile character string handling provides for extracting, inserting, replacing, or moving character strings or substrings of variable lengths.
- Arithmetic and relational operators allow extensive manipulation and comparison of numeric data.
- The IMAGE and FORM statements give users a means of representing input and output data in several formats.
- Programs can be segmented. They can also be developed in a main program/subprograms format.
- The INPUT FIELDS statement and the PRINT FIELDS statement provide for programmed control of terminal input and output.
- Arrays of up to 7 dimensions are allowed.
- Programs can be developed, edited, compiled, debugged, and executed interactively. Debugged programs can be compiled and executed in batch mode.
- A HELP command is available to display information about IBM BASIC commands, statements, etc.
- Users have easy access to system functions, such as MERGE, FETCH, or RUN.
- Some statements can be executed immediately when entered, for tasks such as doing calculations and program debugging.

Programming Language Products

Many built-in (intrinsic) functions are provided. For example, SQR(X) for calculating the square root of X, or TIME\$ for time of day (HH:MM:SS).

Reference

IBM BASIC General Information, GC26-4023, provides a general description of the product.

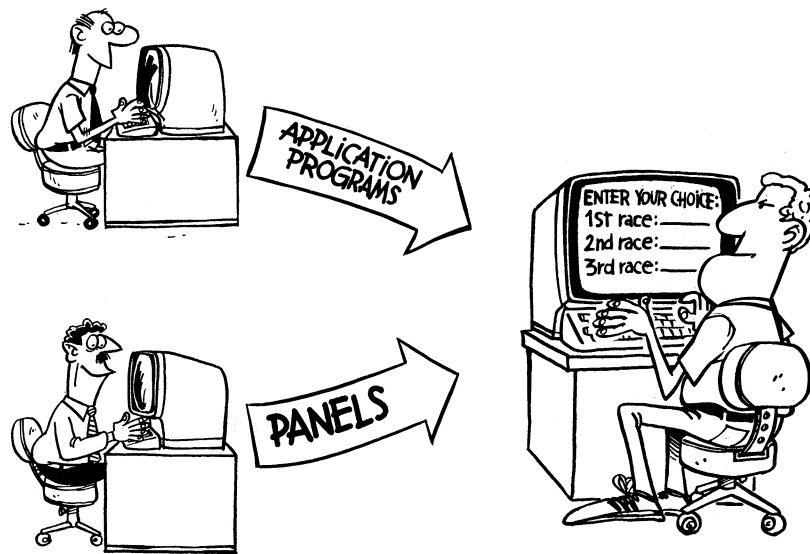
IBM BASIC Application Programming: Guide, SC26-4027, provides guidance for designing, coding, executing, and debugging IBM BASIC programs.

IBM BASIC Application Programming: System Services, SC26-4028, provides information about using the IBM BASIC processor, including related commands.

Display Management System for CMS (DMS/CMS)

Overview

- DMS/CMS can help users format and use displayed panels in an interactive environment.
- Three basic tasks are involved:
 - Designing the panels
 - Programming the applications that use the panels
 - Using the applications.



Many programs use displayed panels as a vehicle for communication between users and the programs. For example, the PROFS facility described in this book uses interactive panels for communicating with users. These panels are often in the form of a menu. A menu allows users to make selections by keying in words in the spaces provided, or by simply pressing a program function key. DMS/CMS is designed to help users create display panels and write the application programs that use them.

Programming Language Products

Panel Creation under DMS/CMS

DMS/CMS makes five display screens available to help create panels for use with application programs. These screens are, in effect, panels themselves. However, DMS/CMS documentation calls them screens to distinguish them from the panels being created. The five DMS/CMS screens are:

- Panel Name - for giving the panel a name.
- Panel Size - for defining the panel's dimensions.
- Design Grid - for designing the panel's layout and content.
- Field Definition - for defining panel field characteristics, such as intensified display, blinking, or color.
- Completion Options - for telling DMS/CMS whether to save the panel and if another panel is to be created.

Movement from screen to screen and cursor movement within a screen is controlled by program function keys. Most information needed to use these screens is included in the information displayed on the screens. However, HELP screens are available to assist users if needed.

DMS/CMS has its own set of editing commands for creating panel designs on the Design Grid screen. The format and content of the information developed on the Design Screen become the format and content of the finished panel.

Application Programs and Procedures for Using Panels

Panels created under DMS/CMS are used with either programs or EXEC procedures developed for that purpose. Programs can be written in COBOL, PL/I, RPGII, or Assembler language. The EXEC procedures can be written in any of the command language processors supported by VM/SP. DMS/CMS makes special commands, macro instructions, and procedures available to aid programmers in writing programs and procedures for display panels.

There are many ways to use DMS/CMS created panels. For example, they can be used to replace traditional paper forms that must be filled in by hand. The panel offers clear advantages over paper forms, such as:

- Ease of filling in data and correcting errors
- Ease of handling, storing, and retrieval
- Ease of changing panel layouts
- Variable number of copies can be requested
- Reduced storage space requirements.

Designers who layout the panels and the programmers who write routines that use the panels can develop a wide variety of applications for using a computer interactively.

Reference

DMS/CMS Guide and Reference, SC24-5198, describes the program and how to use it, and provides examples of related programs in the supported languages.

Chapter 8. Distributed Data Processing Using VM/SP

Distributed data processing (DDP) is data processing in which the processing facilities are spread over multiple sites. Two major factors have contributed to the growth of distributed processing under VM/SP:

- The relative cost of hardware has declined. This encourages the use of additional facilities located where they are most accessible by those who need them.
- VM related licensed programs are easy to use. This means that all data processing work need not be done by experts at the "home office."

The purpose of this chapter is to introduce some of the concepts, facilities, and terminology associated with DDP.

Some DDP Definitions

Terms used in this chapter that might be new to you are:

- Distributed data processing (DDP) — Data processing in which the hardware and software facilities are spread over multiple locations.
- DDP network — A distributed data processing environment in which the multiple locations are connected by telecommunications media.
- Central site — The network site responsible for providing network management and support services.
- Distributed site — In a DDP network, a site that does not participate in overall network management or support services.
- Remote site — From the perspective of any given network site, all other sites in that network are remote sites. The central site is a remote site to each of the distributed sites in a network, and vice versa.
- Link — Facilities for communicating between two adjacent nodes in a network.
- Node — A defined point in a network where one or more links terminate.

Chapter Contents

Why Distributed Data Processing?	223
On-site Data Processing Has Its Advantages	223
Centralized Processing Also Has Advantages	224
A DDP Network Offers the Best of Both Methods	225
The Pieces That Make Up a VM/SP DDP Network	226
System Requirements for the Central Site	227
Resource Requirements for the Distributed Sites	228
Mixed VM SNA and Non-VM SNA Networks	229
Programmable Operator Reduces Network Operator Requirements ...	230
How the Programmable Operator Works	231
The Programmable Operator from the User's Point of View	232

Why Distributed Data Processing?

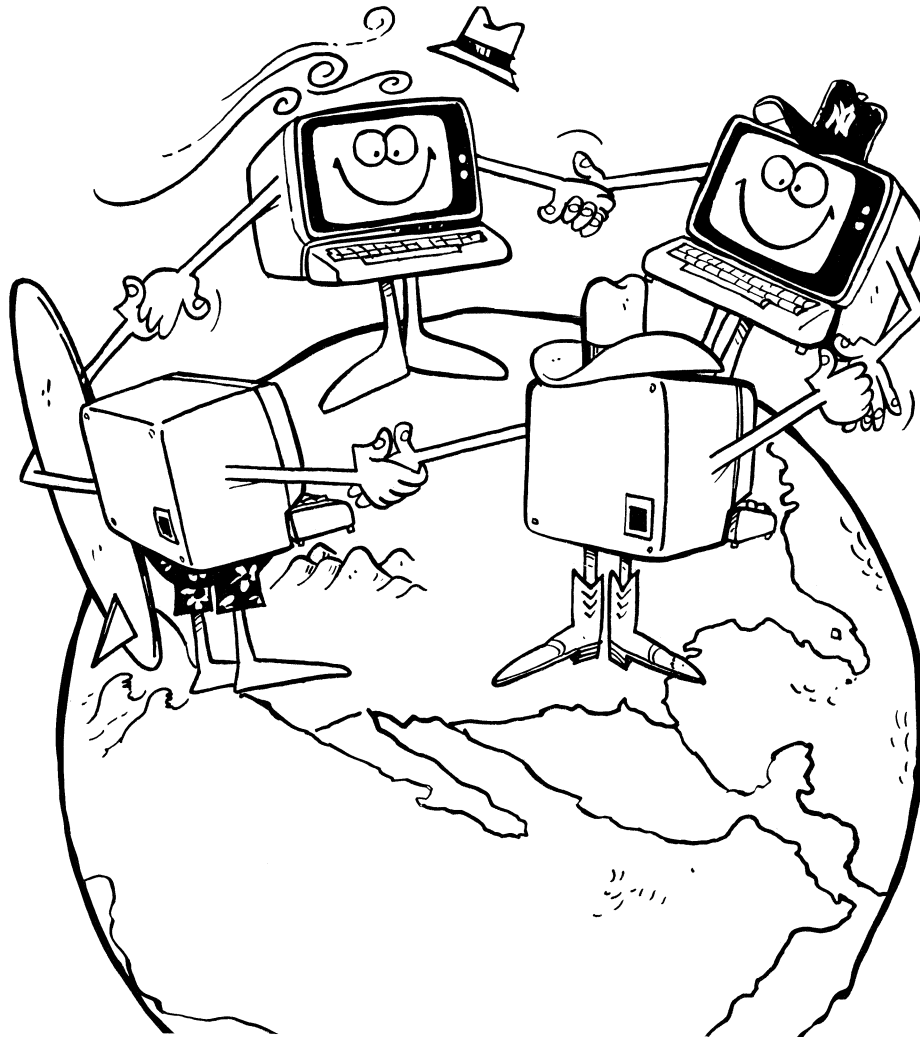
Overview

- Distributed data processing (DDP) is a cost-effective way for a multiple-site enterprise to process data.
- DDP puts facilities close to the people who need them.
- Tasks that require special skills can be performed by support people at a central site.
- Multiple processing sites mean better work flow flexibility.
- A DDP network provides more flexible growth options than central site processing.
- VM/SP supports a wide range of processors that can be used in the DDP environment.
- VM/SP, together with other related licensed programs, provide an excellent, easy-to-use DDP environment.

On-site Data Processing Has Its Advantages

Today it is common for companies to have operating units distributed over an entire country, or even throughout the world. In providing for the data processing needs of such companies, there are advantages to having each location maintain its own processing facilities. For example:

- On-site processing puts facilities close to the people who are most involved with the data being processed.
- The smaller distributed systems can often be designed to fit the needs of local operations better than one central system attempting to satisfy the needs of many locations.
- Distributed systems allow for more flexible growth patterns.
- Delays associated with shipping data to a central facility for processing can be avoided by doing the processing on-site.



Centralized Processing Also Has Advantages

On the other hand, there are also advantages to having data processing for all locations operate as a unit. For example:

- Fewer skilled support people, such as system programmers, serve the needs of all operating units.
- Operating procedures are easier to manage.
- This method reduces dependence on commercial communications facilities.

A DDP Network Offers the Best of Both Methods

The DDP network is a solution that offers the advantages of distributed processing facilities operating as a unit. In addition to all the advantages listed above for both centralized processing and on-site processing, a DDP network offers flexibility of resource use. Workload can be easily transferred from one network site to another. In the DDP network environment the bulk of network management, operation, and servicing can be done efficiently by a relatively few support people at a central site.

VM/SP facilities are available to DDP networks that allow:

- Quick and easy communication among network systems
- Initialization and management of distributed network systems by a single operator
- Diagnosing system problems at a distributed site from a central site
- Logging onto and using distributed network systems
- Operator messages from multiple locations to be handled from one location, some automatically without operator involvement.

Reference

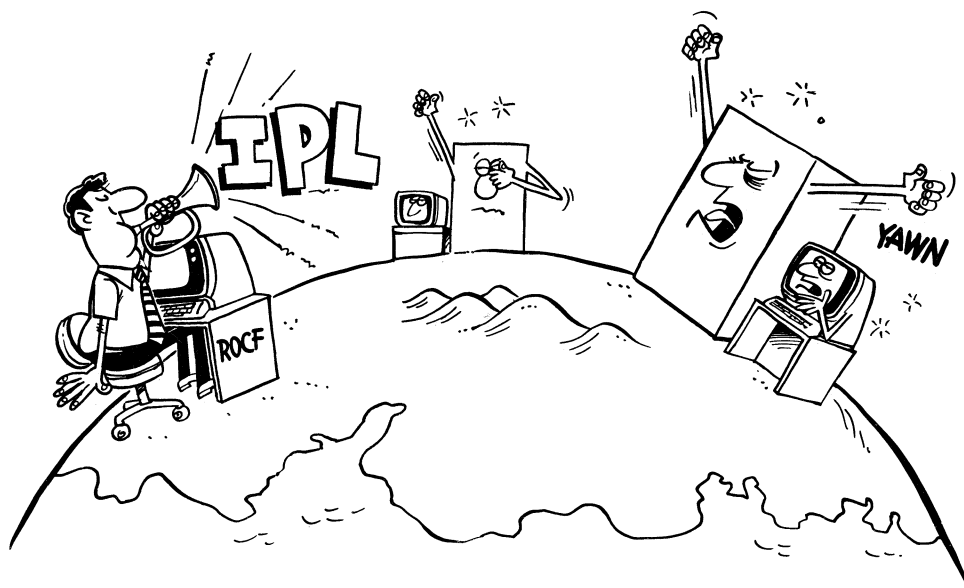
VM/SP Distributed Data Processing Guide, SC24-5241, describes tasks relating to the planning and use of distributed data processing.

The Pieces That Make Up a VM/SP DDP Network

Overview

The basic components of a distributed data processing network are:

- Hardware facilities at the central site and at distributed sites (nodes)
- Programs to manage data flow among nodes
- Transmission media (data links) for carrying data between nodes.



A major advantage of the distributed data processing (DDP) network is that many of the more costly resources, including support people such as system operators and system programmers, are required at only one of the network sites. Facilities are available to allow performance of tasks such as system initialization and problem diagnosis for remote systems. The network location at which the group of network support people work is called the central site. Other locations in the network are called distributed sites.

The central site and the distributed sites both have their own hardware, software, and personnel requirements that must be considered when planning a DDP network. The extent of these resource requirements depends on what the processing capabilities of the network are to be.

Processor features and VM/SP facilities, plus related programs form a foundation for a VM/SP DDP network. These include:

- Processor and VM/SP:
 - IBM 4300 Remote Operator Console Facility (ROCF)
 - Programmable Operator Facility.
- Program products:
 - VM/Pass-Through
 - RSCS Networking
 - Advanced Communications Facility for Virtual Telecommunications Access Method (ACF/VTAM) for VM/SP with VM SNA Console Support (VSCS)
 - Network Communications and Control Facility (NCCF).

Programming requirements for DDP planning are release sensitive. RSCS used here, refers to Version 2 or later.

System Requirements for the Central Site

Tasks performed at the central site in a DDP network, together with any programs and system facilities required to perform these tasks are listed below.

- Do data processing on a local system.

Any VM/SP supported processor might fill this need, but some items in this list require that the processor be an IBM 4300.

- Initialize processors and operating systems at the distributed sites from the central site system.

The distributed processors initialized in this way are IBM 4300 processors with Remote Operator Console Facility (ROCF). The central site operator can initiate this procedure through one of the following:

- An emulated IBM 3275-2 Display Station on the central site processor. Emulation is provided by VM/Pass-Through.
- An IBM 3275-2 Display Station

Each site must have the proper data transmission facilities (for example, modem and transmission lines.)

- Log onto and use systems at other network sites.
 - VM/Pass-Through.
 - ACF/VTAM for VM/SP

- Send certain commands to distributed systems or respond to certain messages that start at distributed sites but must be handled by the central site.

- Programmable Operator Facility
- NCCF
- RSCS Networking

This book describes the Programmable Operator Facility and RSCS Networking in separate topics.

- Transmit messages and files to network sites.
 - RSCS Networking
 - RSCS Networking
- Manage operations, access to resources, and problem data collection in SNA networks.
 - NCCF
- Perform problem diagnostics on distributed site systems.
 - IBM 4300 processor Remote Support Facility (included in ROCF)

Resource Requirements for the Distributed Sites

In the DDP environment, network management tasks are handled at the central site. Therefore, the list of tasks performed by people at distributed sites is understandably shorter than that for the central site. This helps point out the advantage of using systems at multiple locations in a DDP network fashion.

People at distributed sites can:

- Do data processing on a local system.
- Log onto and use systems at other sites on the network.
- Transmit and receive messages and other data over the network.
- Communicate with the central site operator.

The programs and system facilities required at distributed sites are similar to those listed above for corresponding tasks at the central site.

Mixed VM SNA and Non-VM SNA Networks

A mixed SNA network consisting of VM, MVS, and VSE nodes is also possible. These nodes can communicate with each other if they have NCCF Version 2 and ACF/VTAM Version 3 for VM/SP.

Reference

VM/SP Distributed Data Processing Guide, SC24-5241, discusses distributed data processing network requirements.

Systems Network Architecture Concepts and Products, GC30-3072, is an introduction to the benefits and concepts of a Systems Network Architecture (SNA) network. It also covers IBM products that run in the SNA network.

Systems Network Architecture Technical Overview, GC30-3073, has detailed information on the major functions of the SNA network.

ACF/VTAM General Information (for VM/SP), GC30-3246, describes ACF/VTAM, SNA concepts, installing and starting ACF/VTAM, commands, and ACF/VTAM application programs.

Programmable Operator Reduces Network Operator Requirements

Overview

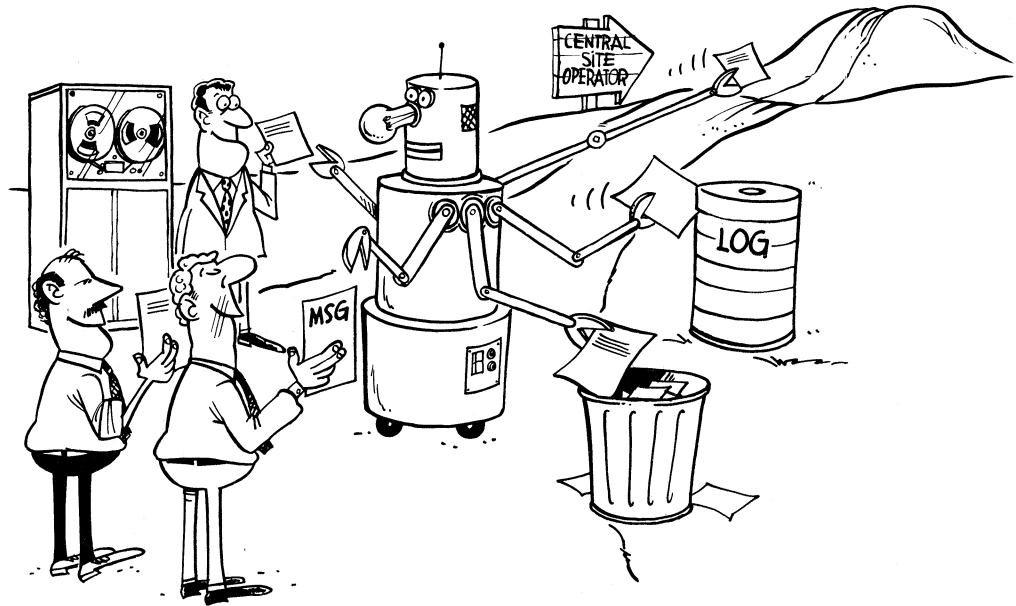
- The Programmable Operator Facility is a part of the VM/SP program product (Release 2 or later). It runs in a CMS virtual machine.
- The programmable operator may be used to increase efficiency in:
 - A single VM/SP system configuration
 - Both the central site and the distributed sites of a VM/SP network
 - The SNA environment with mixed VM, OS/VS, and VSE systems (Release 4 or later.)
- The programmable operator is designed to improve system operation efficiency by:
 - Handling automatically some routine system operator tasks
 - Allowing some operator tasks for distributed systems on a VM/SP network to be done by an operator at a central site.
- The programmable operator uses the facilities of RSCS Networking program product when performing tasks in a VM network environment. The programmable operator also uses the facilities of the NCCF Message Queueing Service when performing tasks in a mixed environment.

The Programmable Operator Facility (programmable operator) can greatly reduce the number of routine tasks performed by the system operator by:

- Logging messages
- Answering certain messages with programmed responses
- Routing certain messages from distributed sites (without operators) to the host site operator for handling
- Executing VM commands.

The programmable operator can be used effectively in a stand-alone system, or as part of network or mixed system management. In a single-system environment it helps relieve the system operator from routine tasks by

filtering out those for which some action is required of the operator, and providing programmed responses for more routine situations.



How the Programmable Operator Works

Some examples of tasks performed by the programmable operator are:

- Determining how incoming messages are to be handled
- Selecting an action routine to handle messages
- Determining user authorization to issue commands through the programmable operator.

The programmable operator must have information available to help it perform its tasks. This includes authorization for users to issue programmable operator commands. The source of this information is one or more CMS files called routing tables. One sample routing table is supplied with Programmable Operator Facility. It can be edited to suit an installation's needs. Others can be created for special purposes.

The programmable operator determines the action to be taken by comparing input messages or commands to information in a routing table. It then invokes the appropriate action routine. The programmable operator has routines to:

- Forward messages to the host system operator
- Execute a CP or CMS command
- Load a new routing table (LOADTBL command).

Other action routines can be written by an installation as needed.

The programmable operator time stamps and stores all messages it receives in a CMS file called the log file. For certain types of messages, it also stores the userid and nodeid of the sender. Authorized users can issue the programmable operator GET LOG command to retrieve the contents of the log file. Messages remain in the log file until purged by an authorized user.

The programmable operator maintains another CMS file, called the feedback file, into which authorized users can store notes. These notes usually relate to system operation, and can be collected by the person responsible for taking appropriate action. The feedback file, like the log file, must be periodically purged.

The Programmable Operator from the User's Point of View

If the programmable operator is to handle messages intended for the system operator, as is commonly the case, it must run in the virtual machine defined for the system operator. However, for other applications, the programmable operator can run in any virtual machine. The programmable operator can be controlled from a different virtual machine on the same system, (using facilities of RSCS Networking program product) from a system at a different network node, or from an NCCF operator station in a mixed environment.

After loading CMS, the system operator can invoke the programmable operator facility by executing the PROPST EXEC or by issuing the PROP command. The central site operator can initiate this action for the central site and distributed sites. When the PROP command execution completes, the programmable operator is ready to use.

The programmable operator processes incoming messages according to entries in the routing table.

1. The programmable Operator at the central site
 - Filters out and logs messages that do not require the system operator's immediate attention. For example, it handles messages that indicate that an operator initiated action completed successfully.
 - Routes certain messages to the person responsible for taking action on those messages. For example, messages that indicate that some action must be taken on an input or output device.
 - Automatically issues appropriate programmed responses to certain situations. For example, relocating information on direct access areas that are beginning to fill.

2. The programmable Operator at distributed sites

- Responds to messages and performs tasks that do not require an on-site operator. For example, it stores user comments in the programmable operator feedback file.
- Filters out and logs messages that do not require the system operator's attention. For example, messages associated with users logging onto and off the system.
- Routes messages that require on-site handling, such as mounting a magnetic tape, to the proper person to take action.
- Routes messages that require handling by central support personnel to the central site. For example, a user at a distributed site requesting the time that his system will be shut down by the central site operator.

After the central system operator has started Programmable Operators at the central and distributed sites, the operator can disconnect the programmable operator virtual machine, and then log onto his working virtual machine.

The TELL facility of CMS can be used to issue VM commands to a programmable operator, either on the same system or on a different system of the same network (through the facilities of RSCS Networking). Commands to be issued are specified as message text on the TELL command line.

In a mixed environment, the NCCF PROP command is used to issue programmable operator and VM commands to the programmable operator from an NCCF operator's terminal. Messages are transferred through the Programmable Operator/NCCF Message Exchange (PMX) part of the programmable operator. Commands to be issued are input as parameters to the NCCF PROP command.

The programmable operator has its own command subset. Authorized (by routing table entries) users can issue these commands to:

- Place comments in the feedback file
- Retrieve the feedback file or the log file
- Load a new routing table
- Change the logical operator assignment
- Write a message to the log file
- Determine the name of the current routing table and the status of logging and node-checking functions

DDP Using VM/SP

- Enter programmable operator debug mode (from programmable operator virtual machine console only)
- Change the status of logging and node-checking functions
- Stop operation of the programmable operator.

In addition, any CP or CMS command can be issued by including it as the operand of the programmable operator CMD command.

Reference

VM System Facilities for Programming, SC24-5288, describes the programmable operator, its routing table, and its action routines. This book also explains how to install and use the programmable operator in single, distributed, and mixed environments.

Chapter 9. Connectivity, Personal Computers, and VM/SP

This chapter describes VM/SP support available for Personal Computers:

- connecting them to a host system
- using them as terminals
- uploading and downloading files
- using host system resources to enhance the Personal Computer

Three products are introduced:

- Enhanced Connectivity Facility
- PC/VM Bond
- VM/PC

Chapter Contents

IBM Enhanced Connectivity Facility	237
Why is there a need for Connectivity Facilities?	237
What Do You Need to Use the Connectivity Facility?	237
What You Can Do with Enhanced Connectivity Facilities	238
Running Enhanced Connectivity Facilities Applications	238
Writing IBM Enhanced Connectivity Facility Applications	239
Reference	240
PC/VM Bond	241
Overview	241
DOS/VM Communications	241
Extra Disk Drives	242
Upload/Download Capabilities	242
REXX/PC	242
Requirements	242
Reference	243
VM/PC (Virtual Machine/Personal Computer)	244
Overview	244
Using Local VM/PC Services	244
Host 3270 Session	244
Requirements	245
Reference	245

IBM Enhanced Connectivity Facility

Overview

- The Enhanced Connectivity Facility is designed to enable work stations and hosts to work together effectively by providing a single interface through which work stations and host systems can communicate.
- Work station users can run the Connectivity Facility applications that use host resources.
- Application programmers can write applications to provide host services to work station users.

Why is there a need for Connectivity Facilities?

Work stations are widely accepted in the marketplace. Work stations have brought computer power to the desk of the end user. Organizations and individuals alike appreciate the convenience, power, decreased cost, and ease of use.

However, even with these advantages, a work station user can be isolated from the rest of the organization. A work station user may also require additional computer services that are available only on a host. Using communications hardware to connect the two worlds of desktop computing and traditional mainframe processing offers a solution to this problem.

IBM Enhanced Connectivity Facilities provide a consolidated approach to work station/host connectivity. It provides a single interface that allows programmers to write work station/host applications for work station users. For the programmer, this means that applications can be written for different work stations and host systems using the same interface, independent of the physical connection of the work station to the host. For the work station user, this means that applications that gain access to host facilities can be run without the need for the work station user to learn all about the host system.

What Do You Need to Use the Connectivity Facility?

The Connectivity Facility is provided on a variety of work stations, host systems, and physical connections. The IBM 3270 Personal Computer and Personal Computer AT ® are just two of the work stations that are supported. Two host systems that provide the Connectivity Facility are VM/SP and TSO Extensions (Release 3 with the MVS/XA feature). For a complete list of the work stations, host systems, and physical connections that are supported and the licensed programs you need to use the Connectivity Facility, see the *Introduction to Enhanced Connectivity Facilities*, GC23-0957.

Connectivity, Personal Computers, and VM/SP

What You Can Do with Enhanced Connectivity Facilities

Enhanced Connectivity Facilities was designed for two distinct users:

1. The work station user who'll be running Enhanced Connectivity Facilities
2. The application programmer who'll be writing the Enhanced Connectivity Facilities applications

Running Enhanced Connectivity Facilities Applications

Before work station users can use any of the services of the Connectivity Facility, communications between the work station and host system must be started. Starting communications requires work station users to start an Enhanced Connectivity Facility program on both the work station and host system. For more information on starting communications on the work station, see *IBM Enhanced Connectivity Facilities PC Requesters User's Guide*, part number 6316993. For more information on starting communications on VM/SP, see the CMSSERV command in the *VM/SP CMS Command Reference*.

After IBM Enhanced Connectivity Facility communications have been started, work station users can run applications that use host resources.

IBM provides the IBM Enhanced Connectivity Facilities as separately orderable licensed programs and also allows programmers to write their own applications. This gives work station users access to some common functions and allows users access to other applications written specifically to their needs.

Some of the functions that are provided with IBM Enhanced Connectivity Facilities include:

- Virtual Disk

Users can use a host system's disk space as if it were a fixed disk on the work station. The data is actually stored at the host and format conversions between host and work station files is done automatically. This means a user can have access to storage beyond the capabilities of the work station.

- Virtual File

Users can use host-formatted files at their work stations. Even though the file exists on the host system in the host's format, a user can use the file at a work station and it will appear as any other work station file. In addition, various format conversions are available for Virtual Files.

- Virtual Print

Users can access host printers allowing them to print documents on high speed devices.

- File Transfer

Users can copy files between their work stations and the host and won't have to worry about translating any data.

- Host Data Base Access

Users can extract data from host data bases and use that data on the work station.

Writing IBM Enhanced Connectivity Facility Applications

Application programmers can write their own applications for work station users by writing a pair of programs called "requester" and "server" programs.

A requester is a program that requests a host service.

A server is a program that performs one or more service functions on behalf of a requester.

Typically, requesters are programs that run on the work station and servers run on the host.

How do these requester/server pairs communicate with one another? IBM Enhanced Connectivity Facilities provide an interface through which requesters and servers can communicate. The interface is the Service Request Programming Interface (SRPI).

The SRPI:

- Provides a consistent interface for requesters to obtain services and/or data from different hosts
- Isolates requesters and servers from underlying communications environments on work stations and hosts
- Enables applications to be written for a number of different work station and host systems.

By creating requesters and servers that communicate through the SRPI, application programmers can provide services to work station users from the host. This shields the work station user from the host and avoids forcing the user to learn all about the host system, yet gives the user access to the power of the host system.

For example, suppose that on a particular VM/SP system, an application programmer has access to a network directory. This directory contains the

name, address, phone number, userid, and node of every VM/SP user in the network. By writing a requester for work stations and a server on VM/SP, the application programmer can give work station users or programs access to this directory. The application can be designed to shield work station users from VM/SP and still provide directory information to the users.

Reference

Introduction to IBM Enhanced Connectivity Facilities, GC23-0957, discusses the Connectivity Facilities in greater detail.

IBM Enhanced Connectivity Facilities Processing Services PC Requesters User's Guide, part number 6316993, discusses the use of the Connectivity Facility on work stations in detail.

VM/SP CMS Command Reference, SC19-6209, contains information on how to start IBM Enhanced Connectivity Facilities processing on VM/SP using the CMSSERV command.

VM/SP Programmer's Guide to the Server-Requester Programming Interface for VM/SP, SC24-5291, discusses and gives examples of how to write a server.

PC/VM Bond

Overview

PC/VM Bond provides connectivity between a Personal Computer and VM.

PC/VM Bond gives the user access to the larger operating system with its built-in advantages of computing power and additional disk storage space. It provides access to the many facilities available on the VM/SP system while retaining the stand-alone advantages of the Personal Computer DOS (Disk Operating System).

With PC/VM Bond, the Personal Computer can be used both as a Personal Computer and as a Display Station connected to a VM System. The user can switch back and forth from VM (or "terminal" mode) to DOS ("Personal Computer" mode) by pressing Alt-Esc. The Personal Computer display is supported as an emulated IBM 3278 Model 2 Display Station.

In terminal mode, the user has access to all the VM commands, macros, programs, etc. In Personal Computer mode, the user has all the functions of DOS plus the added functions of VM/PC Bond:

- DOS/VM communications

- extra disk drives

- upload/download capabilities

- REXX/PC EXEC language

DOS/VM Communications

You don't have to switch back to VM from DOS to communicate with VM. You can send and receive messages to/from other VM users. The PC/VM Bond command, TELL, allows you to send messages to other VM users. When another VM user sends you a message, your Personal Computer will sound three beeps. Pressing the Alt, Left Shift, and Ctrl keys allows you to view the message.

You can enter VM/SP commands from DOS and receive their results on DOS using the advanced features of PC/VM Bond. The Host Programming Interface feature of PC/VM Bond supports this function.

Connectivity, Personal Computers, and VM/SP

Extra Disk Drives

PC/VM Bond gives you the ability to use more disk drives than you physically have on your PC. The extra disk drives are actually VIRTUAL disk drives that reside on your disk on VM. You create a virtual disk using the PCDISK CREATE command. Then you must map the virtual disk to a drive letter for your PC. You map a virtual disk using the SETDISK command. Once you've created and mapped the virtual disks, you can use them just as you would your real disk drives on your Personal Computer.

Upload/Download Capabilities

You can use VM/PC Bond to copy files from VM to your Personal Computer or from your Personal Computer to VM. This process has the appearance of copying virtual disk files to and from real disk files. You use the PCDISK INSERT and PCDISK EXTRACT command to do this. INSERT and EXTRACT copy the file to/from the virtual disk to your VM storage. Another copy to/from your virtual disk to your real PC disk results in the upload/download operation.

REXX/PC

You can use the REXX/PC EXEC language to write programs on your Personal Computer. REXX/PC is a programming language designed for Personal Computer users that is similar to the Command Language Processor (REXX) available on VM/SP. It provides an alternative to the batch facilities in DOS. You do not need the services of VM/SP to use REXX/PC.

Requirements

The following are required to support this connectivity:

- An IBM Personal Computer, XT, XT/370, AT, or AT/370
- An IBM 3278/79 Emulation Adapter Card or the IBM 3278/79 Advanced Emulation Adapter Card
- DOS Version 2.00 or later
- PC/VM Bond, which is made up of two parts:

VM Bond, which operates on the Personal Computer.

PC Bond, which operates on your VM/SP system. Your system administrator is responsible for installing and giving you access to PC Bond.

- And, of course to use VM/SP, you'll need a userid, password, and an A-disk on your VM/SP system.

Reference

PC/VM Bond User's Guide, part number 6317007, teaches you how to use PC/VM Bond. It contains a primer that introduces the user to PC/VM Bond and a reference section with all the commands.

PC/VM Bond Programmer's Guide, SH24-5087, tells you how to plan and install PC/VM Bond.

PC/VM Bond Quick Guide, SX24-5158, is a summary of information needed to perform most of the functions provided by PC/VM Bond.

VM/PC (Virtual Machine/Personal Computer)

Overview

VM/PC is a DOS application that runs on an AT/370 and gives you a VM system within a personal computer.

VM/PC enables you to:

- Use a local 370 environment in a PC AT/370. This allows VM/PC users to run CMS applications in their personal computers.
- Emulate a 3278/79 terminal.
- Export and import files to and from DOS.
- Maintain active sessions between DOS, VM/PC, and a host system.
- Transfer data and programs between your host VM computer and your IBM Personal Computer.

Using Local VM/PC Services

You can connect to the local computer in your IBM Personal Computer and run the local VM/PC system. You can authorize several people to use VM/PC, but because there's only one machine, only one person can use it at a time. If many users have access to the machine, you can control access to VM/PC and to each user's data. The VM/PC Configurator controls who has access to what and is described in the *Installing and Configuring VM/PC* book.

Host 3270 Session

You can connect to a host computer as a 3270 terminal. You can combine your host VM session with your local session into a single environment. This allows you to use some of the resources of your host system as if they were attached to your local VM/PC session. You can:

- Access CMS minidisks on your host VM system. This allows you to:
 - run programs on minidisks attached to your host system
 - copy files from your host system to your local system and from your local system to your host system.
- Print files on your host VM system's printers

The VM/PC Host Server allows you to spool your print output to your host virtual printer so any printing done on your local session will be

sent to your host virtual printer. This gives you access to high-speed printers.

- Access your host VM system's reader, punch, and console

Your IBM Personal Computer does not support real or virtual readers or punches; however, the VM/PC Host Server allows you to use your host VM devices as if they were local devices.

- Transfer DOS files to your host VM system

You can transfer DOS files to your local VM/PC system by using the VM/PC IMPORT command. You can transfer VM files to DOS by using the VM/PC EXPORT command.

Requirements

To use VM/PC, you need:

- an IBM Personal Computer AT/370
- one fixed disk (minimum)
- one diskette drive (minimum)
- VM/PC 2.0
- DOS Version 3.10

To use VM/PC and VM/PC Host Server, you need:

An IBM 3278/79 Emulation Adapter Card

Coaxial connection for the 3278/79 Emulation Adapter Card

Reference

VM/PC User's Guide, SC24-5254, teaches you how to use VM/PC.

VM/PC Host Server Programmer's Guide, SH24-5125, contains the information needed to plan and install the VM/PC Host Server.

Chapter 10. VM/Integrated System

This chapter discusses VM/Integrated System.

VM/IS provides a powerful and responsive computing system for Engineering/Scientific and Business professionals. It is a general purpose system designed to help improve the productivity in many of your organization's daily activities.

Chapter Contents

Creating Documents	249
Creating Graphics	250
Engineering/Scientific	250
Administrative	250
Communications	251
Data Management	251
Business Computing	251
Intelligent Workstation Support	252

Overview

VM/Integrated System is based on VM/SP. It is a *preconfigured* VM/SP. This preconfigured concept frees you from the complex task of designing your configuration, reducing the skill level and time you require to install your system.

VM/IS also has several optional packages. Each optional package contains a more specialized product set geared to a specific area of use. This modular design allows you to add any combination of these optional packages to your base. The optional packages act as building blocks allowing you to select only those products you require.

Installing VM/IS

VM/IS's predesigned software is packaged with its own installation procedure. This procedure is designed specifically for VM/IS thereby reducing the number of steps and decisions you must make to install the system.

VM/IS is a preconfigured system. It supplies a number of these configurations for you. During the installation process, you choose one of these configurations.

The VM/IS documentation walks you through the installation process. It gives step-by-step instructions that tell you what to do and how the system will respond.

VM/IS makes its products available to you through the use of a menu interface. It describes the tasks you can perform using VM/IS. This interface is a series of panels and menus that appear on your terminal display. These panels and menus help you select the correct product to complete a specific task.

Using the Base Products

The base products allow you to create documents, produce graphics, manage your files, and submit and monitor batch jobs.

Creating Documents

VM/IS provides tools designed to help you create documents without having to manually format the text.

You create text files that contain both the text of the document and *markup* that describes how you want the document to look. You then format the file using a text formatter which produces the finished product. You can send the formatted document to a printer or distribute the document to other users for review.

VM/Integrated System

Creating Graphics

VM/IS also provides tools to help you produce a variety of graphics charts.

Using the Optional Packages

The optional packages provide various sets of software products that you can use to perform many different tasks. The packages are modular, allowing you to purchase any combination of them.

Engineering/Scientific

VM/IS offers an Engineering/Scientific Program Development Support (E/SPDS) package to aid the engineer or scientist. It includes:

- program development facility:
 - create and edit programs
 - store and retrieve programs
 - compile and execute programs
 - perform general maintenance on program libraries.
- VS FORTRAN programming language
- high accuracy arithmetic subroutine library
- elementary math library
- FORTRAN utilities.

Administrative

The administrative tools provided by VM/IS reduce your dependence on mail, telephone, and other conventional services.

- schedules - you can keep and update your personal calendar, schedule conference rooms and equipment, and group meetings.
- reminders - you can have the system remind you about meetings, appointments, things to do.
- mail - you can read and handle mail that other users have sent to you on the system.
- notes and messages - you can send notes and messages to other users, and work with notes in your personal storage.
- documents - you can write and edit letters and other documents.

Communications

You can use VM/IS to create a computer system network. Such a network can enhance your communication with other sites in your organization.

Rather than send letters and documents using conventional mail systems, you can transmit your letters and documents electronically to systems at other sites using the tools of VM/IS.

VM/IS also provides tools to allow you to log on and use remote systems as if your computer terminal were directly attached to the remote system.

Such a capability is useful if you are visiting a different branch of your organization. To maintain close communications with your home office, you can log on to your home system through the remote system in the branch you are visiting.

Data Management

VM/IS provides data base management tools to help you manage your data. With these tools, you can create and maintain a relational data base. You can:

- Create a data base to allow the sharing of data and to improve data integrity and control
- Retrieve data from the data base
- Produce reports in several formats
- Perform calculations on the data to reflect totals, subtotals, averages, maximums, and minimums in your reports
- Add, delete, or change data in the data base.

Business Computing

VM/IS provides programming languages you can use to help solve everyday business problems:

- VS APL
- PASCAL/VS
- BASIC

VM/Integrated System

Intelligent Workstation Support

VM/IS provides support for the IBM Personal Computer (PC). You can use the following PC software products with VM/IS:

- Personal Computer/Virtual Machine Bond (PC/VM Bond)
- Virtual Machine/Personal Computer (VM/PC)
- 3270 Personal Computer File Transfer Program (PC File Xfer).

PC/VM Bond allows regular PC users to take advantage of some of the key features of VM/IS without having to learn VM/IS. VM/PC allows regular VM/IS users to use a PC as a stand-alone VM/IS system.

References

VM/IS General Information , GH24-5119, contains more about what VM/IS is, what it can do, and how it can help your organization.

Chapter 11. Planning, Installing, and Servicing VM/SP

This chapter is an overview of three tasks that at some time apply to every VM/SP installation:

- Planning for the system
- The installation process
- Applying service updates to the system.

You should use the *VM/SP Planning Guide and Reference* and the *VM/SP Installation Guide* for specific information on planning, installing, and servicing VM/SP. This chapter highlights the contents of those two books. You should not use this chapter for guide or reference information.

Chapter Contents

Planning VM/SP	255
Preparing the Files That Describe Your VM/SP System	255
Planning Storage Requirements	256
Installing VM/SP	258
Servicing VM/SP	260
Corrective Service	260
Preventive Service	261
Other IBM Supplied Updates (ZAPs, over the phone updates) ...	261
Local Updates	261

Planning VM/SP

Overview

Among the things that must be considered when planning for VM/SP installation are:

- System control files
- Storage requirements
- Device requirements
- Minidisk allocation
- Options that affect performance
- System libraries
- Programming language support
- Access method support
- Virtual machine operating systems other than CMS
- Special situations.

One purpose of this topic is to make you aware that planning is an important part of the VM/SP installation process. The better a VM/SP installation process is planned, the less time it takes to do the job. If properly done, the actual installation takes far less time than the planning process.

Before installing VM/SP there are many things that must be thought out. In addition, there are things that can be done in advance that make the installation proceed more smoothly. This topic briefly discusses a few of the more important aspects of system planning. Anyone who is to take part in the planning process for a VM/SP installation should also read *VM/SP Planning Guide and Reference*.

Preparing the Files That Describe Your VM/SP System

A major task in getting ready to install VM/SP is the preparation of files that define your VM/SP system. These files are:

- System directory - defines virtual machines for all authorized users.
- DMKRIO - defines the real system input and output devices, control units, and channels.

Plan, Install, and Service

- DMRSYS - defines many of the CP system control functions.
- DMRSNT - identifies saved systems for installations that plan to use them.

Depending on the installation method used, these files can be ready to install at system generation time. For example, you can prepare control statements for use in creating the system directory in advance. You can plan for tailoring the copies of system files supplied with VM/SP so that they require minimum time during installation.



Planning Storage Requirements

Planning for VM/SP system storage requirements falls into two general categories:

- Real storage planning
- Direct access storage planning.

Many factors affect how much of each kind of storage any given system might require.

Some of the factors that affect real storage requirements are:

- Nucleus size
- Number of system real devices, control units, and channels

- Types of system devices
- Number of virtual machines defined in the system directory.

Some of the factors that affect direct access storage requirements are:

- Nucleus size
- Maximum number and size of logged on virtual machines
 - Paging space
 - Spooling space
 - CMS minidisks.
- Operating systems running in virtual machines
- Other facilities
 - Error recording
 - System restart
 - System directory
 - System Profile
 - Interactive Problem Control System
 - Saved systems
 - Number of National languages available on the system.

Reference

VM/SP Planning Guide and Reference, SC19-6201, describes how to get ready to install VM/SP.

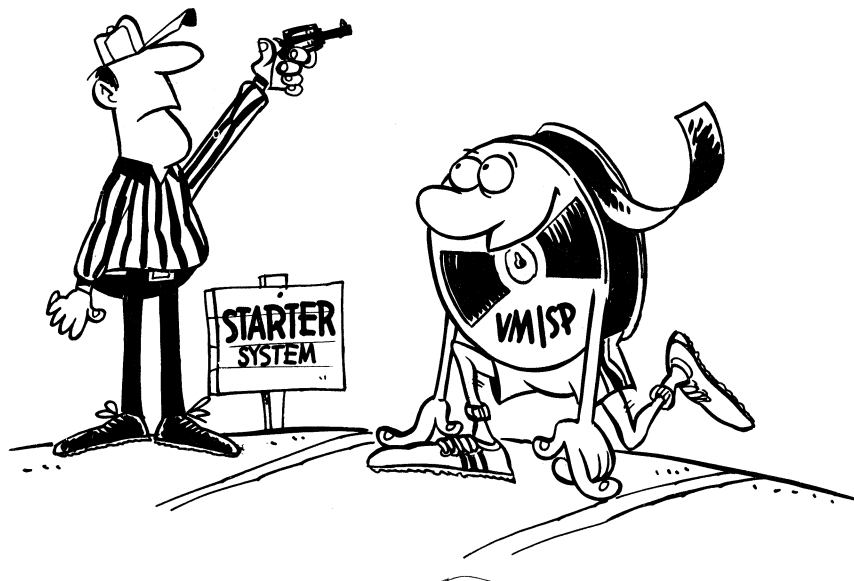
Installing VM/SP

Overview

- There are two kinds of VM/SP installation procedures:
 - Building a new system using a starter generating system shipped with VM/SP.
 - Upgrading to a new level, using an existing system as the generating system

Use of a starter system is the recommended procedure.

- The most important part of any system generation is proper planning before you begin.



The purpose of this topic is to make you aware of what the major system generation tasks are. For information beyond that, you should see *VM/SP Installation Guide*. If your responsibilities include VM/SP installation you should become familiar with the contents of that book. Although the installation of VM/SP is not difficult, there is considerably more to it than outlined in this topic.

Major steps in generating a VM/SP system using a starter system are:

- Plan for required system files.
- Format direct access storage and load the starter system tape to it.
- Prepare system residence direct access storage space and load the VM/SP product tape to it.
- Build the system directory and other required files.
- Generate a CP nucleus.
- Generate a CMS nucleus.
- Generate a GCS nucleus (if desired.)
- Generate a TSAF load module (if desired.)
- Run the installation verification procedures (IVP).

Major steps when using an existing system as the generated system are:

- Load CP and CMS from product tape.
- Generate a new CP and CMS nucleus for the existing system.
- Generate a GCS nucleus (if desired.)
- Generate a TSAF load module (if desired.)
- Verify the installation (IVP).

Reference

VM/SP Installation Guide, SC24-5237, describes in detail the procedures for installing VM/SP.

Plan, Install, and Service

Servicing VM/SP

Overview

You should be familiar with these concepts before you attempt to service any of the VM/SP components installed on your system.

When you have a servicing task, it is important for you to be able to distinguish between the types of code you are working with. For example, some components of VM/SP are maintained as source code. You service source code by updating source statements in an assembly file and assembling that file to create a new executable module. Other components are maintained as object code. You service object code by replacing a preassembled executable module or TEXT deck.

In both cases, the result of service is the same; you have an executable version of the code. However, there are significant differences in the build processes you must go through to make the code executable. (The build processes use service execs and commands).

Types of Service

In addition to the concept of source and object code, you need to know there are different types of service. They are:

- Corrective Service
- Preventive Service
- Other IBM supplied updates (ZAPs, over the phone updates)
- Local updates.

Service should be applied in a test environment before you install it on your production system. Also make sure you have a good back-up copy of the production system, specifically the files being changed by the service.

Corrective Service

At your request, IBM sends you tapes with changes you can incorporate into the code of a product. These tapes contain updates to correct a problem you are having with a product. These changes are referred to as corrective fixes.

The *VM/SP Installation Guide*, contains details on applying corrective service to both source and object code.

Preventive Service

IBM may send you tapes containing replacement files for a product. The replacement files have changes already incorporated in them. Tapes containing this service are called preventive service tapes. For IBM products, these tapes are known as Program Update Tapes (PUTs).

Preventive service is cumulative. When you apply preventive service, you load down the contents of the PUT to minidisks. All the changes on the PUT, except the ones you want to exclude, are applied to the system.

For preventive service, you can use either the product specific service exec or VMSERV. If you have preventive service not contained on a PUT, then you need to invoke each product's specific service exec directly. For preventive service contained on a PUT, use the VMSERV EXEC.

VMSERV:

- Controls positioning of the tape
- Verifies the service level being applied
- Prints the PUT DOCUMENT and Memo(s)-to-Users (optional)
- Executes each product's service EXEC to load service
- Restarts to continue loading service
- Executes the service exec with the option to build.

Other IBM Supplied Updates (ZAPs, over the phone updates)

This type of service usually involves direct over the phone assistance from the IBM support center. You may also need to apply updates to your system when other methods are not available. This involves you creating your own control files, assigning minidisks for your updates, and selecting the appropriate build procedure.

Local Updates

Local updates are user modifications to a component's source code. The procedures that you can use to apply local updates are similar to source updating CP and CMS. The difference is that you must create your own control and auxiliary control files for applying a user modification. Figure 34 below shows the EXECs available for updating VM/SP.

Program	Comments
ASMGEND	Updates the VM/370 system assembler using GENMOD.
CMMSGEND	Creates a new CMS command module from updated TEXT files using GENMOD.

Figure 34 (Part 1 of 2). Programs for Servicing VM/SP

Plan, Install, and Service

Program	Comments
DCSSGEN	Rebuilds the CMSINST discontinuous saved segment from updated TEXT files.
DOSGEN	Rebuilds the CMSDOS and INSTVSAM discontinuous saved segments from updated TEXT files.
GENMOD	Creates a nonrelocatable module file on a CMS disk.
GENTSAF	Rebuilds the TSAF load module from updated TEXT files using a loadlist (e.g. ATSLOAD EXEC) as input.
PRELOAD	Collects multiple TEXT files and reformats them into a single TEXT file.
SAMGEN	Rebuilds the CMSBAM discontinuous saved segments from updated TEXT files.
SPGEN	Does various system generation and maintenance functions, using the parameters contained in SPGEN PROFILE.
VMFASM	Creates an updated source file using IBM updates, PTFs, and user updates, then assembles the source file.
VMFLOAD	Creates a new CP or CMS nucleus based on a control file and a loadlist EXEC file.
VMFLKED	Invokes the CMS LKED command to link-edit modules into a loadlib
VMFMAC	Creates macro libraries using IBM and user updates.
VMFNLS	Applies updates to national language files and compiles the updated versions.
VMFREMOV	Removes PTFs that were applied using VMFMERGE.
VMFTXT	Creates text libraries using IBM and user updates.
VMSERV	Controls the individual service EXECs on the system Program Update Tape.
VRSIZE	Generates DMKSLC text which is used to generate a V = R area in the system.
VSAMGEN	Rebuilds the CMSVSAM and CMSAMS discontinuous saved segments from updated TEXT files.
VSEVSAM	Builds a VSE/VSAM MACLIB containing the supported VSE/VSAM macros.
ZAP	Modifies or dumps MODULE, LOADLIB, or TXTLIB files.
ZAPTEXT	Modifies or dumps individual TEXT files.

Figure 34 (Part 2 of 2). Programs for Servicing VM/SP

Note: There are other service procedures available to service System Network Architecture (SNA) products, namely VMFMERGE, VMFREMOV, and VMFZAP. For details about servicing procedures, see the *VM/SP Installation Guide*.

Reference

VM/SP Installation Guide, SC24-5237, describes procedures for applying service to VM/SP.

Chapter 12. Where to Find More Information about VM/SP

This chapter lists abstracts of the books in the VM/SP library. Charts show how the books relate to general tasks. To find specific information in the VM/SP library, use *VM/SP Library Guide, Glossary, and Master Index*, GC19-6207.

For descriptions of programs that run on VM/SP, see the references cited in related topics in this book. For a complete list of IBM documentation, see *IBM System/370 and 4300 Processors Bibliography*, GC20-0001.

Chapter Contents

Application Development Guide, SC24-5247	267
CMS Command Reference, SC19-6209	267
CMS for System Programming, SC24-5286	267
CMS Primer, SC24-5236	267
CMS Primer for Line-Oriented Terminals, SC24-5242	267
CMS User's Guide, SC19-6210	267
CP Command Reference, SC19-6211	268
CP for System Programming, SC24-5285	268
Distributed Data Processing Guide, SC24-5241	268
EXEC 2 Reference, SC24-5219	268
System Facilities for Programming, SC24-5288	268
General Information, GC20-1838	269
Group Control System Command and Macro Reference, SC24-5250	269
Programmer's Guide to the Server-Requester Programming Interface for VM/SP, SC24-5291	269
Installation Guide, SC24-5237	269
CMS Macros and Functions Reference, SC24-5284	269
Operator's Guide, SC19-6202	269
Planning Guide and Reference, SC19-6201	270
Release 5 Guide, SC24-5290	270
RSCS Networking Version 2 General Information, GH24-5055 ...	270
Running Guest Operating Systems, GC19-6212	270
System Messages and Codes, SC19-6204	270
System Messages Cross-Reference, SC24-5264	270
System Product Editor Command and Macro Reference, SC24-5221	271
System Product Editor User's Guide, SC24-5220	271
System Product Interpreter Reference, SC24-5239	271
System Product Interpreter User's Guide, SC24-5238	271
Terminal Reference, GC19-6206	271
Transparent Services Access Facility Reference, SC24-5287	271

These abstracts are to help you select the book(s) best suited as reference information for your needs. The titles are listed alphabetically.

Application Development Guide, SC24-5247

This book tells FORTRAN and COBOL application programmers how to compile, link, load, run, test, and debug programs using CMS. It also contains information on using the Interactive System Productivity Facility (ISPF) for dialog management and the Structured Query Language/Data System (SQL/DS) for data base management.

CMS Command Reference, SC19-6209

This book provides users of the Conversational Monitor System (CMS) component of VM/SP with detailed reference information about command syntax and usage for: CMS commands; EDIT subcommands; HELP format words; DEBUG commands; EXEC control statements, special variables, and built-in functions.

CMS for System Programming, SC24-5286

This book provides system programmers with information about CMS. This information includes details on interrupt handling, CMS storage maps, updating source programs, assembling and executing programs, creating commands and message files, CMS external macros, programming support for OS and VSE, access method support, CMS support of OS and DOS VSAM functions, and the CMS Batch Facility.

CMS Primer, SC24-5236

This manual teaches you, as a new user, how to do your work using VM/SP and a video display terminal. The primer presents only a subset of all the functions and commands available with VM/SP. The material is presented in an interesting manner with suggested exercises included in the text.

CMS Primer for Line-Oriented Terminals, SC24-5242

This manual is an interactive tutorial for VM/SP users of line-oriented (line mode) video display terminals. The manual, which is similar in scope and content to the *VM/SP CMS Primer*, is designed to quickly give the reader a working knowledge of VM/SP. Topics include logging on; editing, managing, and printing files; using the Document Composition Facility (SCRIPT/VS) to format files; and writing EXECs.

CMS User's Guide, SC19-6210

This book contains general information on the Conversational Monitor System (CMS) component of VM/SP. It provides information and examples regarding the CMS file system, the CMS batch facility, the HELP facility, and CMS Session Services. Also included are examples on using the System Product Editor to create and edit CMS files and using the System Product Interpreter to create and use EXECs.

More Information

CP Command Reference, SC19-6211

This book is a reference manual for all privilege classes that are running systems such as OS, OS/VS, DOS, DOS/VS, VSE systems, CMS and networking systems in a virtual machine under VM/SP. Control Program (CP) commands available to all privilege classes are listed alphabetically. Each command description contains general usage information, the command line format, descriptions of all allowable operands, and default values for operands. Also included are tables showing the relationship of the general class of CP commands to the complete set of VM/SP CP commands.

CP for System Programming, SC24-5285

This manual is a reference for system programmers, system analysts, and others who implement and extend the functions of the Control Program (CP) of IBM's Virtual Machine/System Product (VM/SP). It assumes some experience with programming concepts and techniques. The book has of three parts:

Part 1 describes the functions of CP and provides guidance in using some CP features.

Part 2 describes options available in VM/SP to analyze and improve the performance of virtual machines and operating systems.

Part 3 is a reference for dealing with processor features and real peripheral devices.

Distributed Data Processing Guide, SC24-5241

This book describes the concepts and facilities of distributed data processing when using VM/SP and related programs.

EXEC 2 Reference, SC24-5219

This book defines the EXEC 2 language. It contains all the formats, syntax rules, and descriptions of the arguments for EXEC 2 statements. An EXEC 2 primer for new users is included. It summarizes the language and what it is capable of. A detailed discussion of the different types of EXEC 2 statements is followed by examples. It lists the error messages and return codes issued by the EXEC 2 interpreter.

System Facilities for Programming, SC24-5288

This book provides system programmers with detailed information on facilities available in VM/SP. These facilities include the DIAGNOSE instruction, the Inter-User Communications Vehicle (IUCV) for CP and CMS, CP System Services, the Virtual Machine Communication Facility (VMCF), the programmable operator facility, and information on getting national languages on your system.

General Information, GC20-1838

This book introduces and describes the features and facilities of VM/SP, and provides customer management and technical staffs with information needed to evaluate the applicability of VM/SP to their installations. It contains information needed for a basic understanding of using, programming, and installing VM/SP. It also lists some program products that work with VM/SP.

Group Control System Command and Macro Reference, SC24-5250

This book provides detailed reference material that describes the functions and use of all macros supported in the Group Control System. Each macro description contains information on general usage, format, all available parameters, messages, and return codes. The manual is intended for personnel who are developing application programs to run on the Group Control System.

Programmer's Guide to the Server-Requester Programming Interface for VM/SP, SC24-5291

This book provides an application programmer with information on how to write and install IBM System/370 to IBM Personal Computer Enhanced Connectivity Facilities servers in a VM/SP system. For the work station (for example, IBM Personal Computer) user, the manual also gives information on how to start IBM System/370 to IBM Personal Computer Enhanced Connectivity Facilities communications on VM/SP.

Installation Guide, SC24-5237

This book is for system programmers and administrators who install or apply service changes to VM/SP. It contains step-by-step procedures for installing VM/SP. In addition, there are procedures for applying preventive and corrective service.

CMS Macros and Functions Reference, SC24-5284

This book provides CMS users with detailed reference information about CMS assembler language macro instructions and CMS functions.

Operator's Guide, SC19-6202

This book is intended for those responsible for the operation and administration of VM/SP. It includes descriptions of all those commands that affect the I/O resources and operating characteristics of VM/SP, the associated virtual machines, and the real hardware configuration. Also included is information on spooling, resource allocation, system startup and shutdown procedures, and VM/SP service programs.

More Information

Planning Guide and Reference, SC19-6201

This book is for those people responsible for the planning, installation, and updating of a VM/SP system. This publication includes information about:

- Planning for system generation
- Defining your VM/SP system
- Generating a 3704/3705/3725 control program that runs with VM/SP
- Updating VM/SP.

The reader is expected to have a general understanding of data processing and teleprocessing techniques.

Release 5 Guide, SC24-5290

This book describes the functional changes to VM/SP for Release 5. It includes details for migrating from VM/SP Release 4 to VM/SP Release 5.

RSCS Networking Version 2 General Information, GH24-5055

This book provides an overview of the Remote Spooling Communication Subsystem Networking program product (RSCS). It includes a functional description, discussion of requirements for using the product, and a program summary.

Running Guest Operating Systems, GC19-6212

This book is for people who are interested in running their operating system (VSE, MVS and VM/SP) under the VM/SP host system. The book presents the basic processes so that the working system can be set up quickly. Experienced persons can improve the efficiency of their installation, running a guest system under VM/SP, with the included recommendations. The book assumes the audience is knowledgeable about their own system but new to VM/SP.

System Messages and Codes, SC19-6204

This book contains messages and codes, as well as restrictions, that may be encountered when using the Conversational Monitor System, Control Program, Interactive Problem Control System, Group Control System, and Transparent Services Access Facility. Conditions that generate these messages and codes are explained, the resulting action is described, and appropriate responses are suggested.

System Messages Cross-Reference, SC24-5264

This manual contains cross-reference information for the messages in the manual, *VM/SP Messages and Codes*, SC19-6204. The cross-reference information relates messages to the commands that caused them to be issued, lists messages alphanumerically by message identifier, relates messages to the module that issued them, and lists the messages in alphanumeric order by message text. The manual consists of four appendices, one for each of the VM/SP components: Conversational

Monitor System, Control Program, Interactive Problem Control System, and Group Control System, and Transparent Services Access Facility.

System Product Editor Command and Macro Reference, SC24-5221

This publication is a reference manual that contains all the command formats, syntax rules, and operand and option descriptions for the XEDIT subcommands and macros. It tells how to enter XEDIT commands, subcommands and macros. It contains the format description, and operand and option list for the XEDIT command, which is used to invoke the editor. It lists EDIT subcommands and their XEDIT counterparts. It tells how to define windows and virtual screens used with the CMS Session Services. You should be familiar with the information in the *System Product Editor User's Guide* before attempting to use this manual.

System Product Editor User's Guide, SC24-5220

This book is intended for the individual who has limited data processing experience. It is designed to give the user a working knowledge of the System Product Editor (invoked by the XEDIT command). XEDIT provides a wide range of functions for text processing and programming development. Both a full screen and a line mode editor, it can be used on display and typewriter terminals.

System Product Interpreter Reference, SC24-5239

This book provides statement syntax, format, and usage notes for using Restructured Extended Executor (REXX), the command language for writing procedures to run under System Product Interpreter, the command language processor of CMS.

System Product Interpreter User's Guide, SC24-5238

This tutorial book discusses and provides examples of how to use Restructured Extended Executor (REXX), the command language for writing procedures to run under System Product Interpreter, the command language processor of CMS.

Terminal Reference, GC19-6206

This book discusses some of the various devices that can be used as VM/SP input or output terminals. It includes descriptions of physical characteristics and examples of how to use the terminals.

Transparent Services Access Facility Reference, SC24-5287

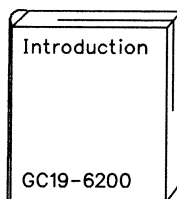
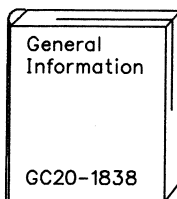
This manual contains all the reference material required to use the Transparent Services Access Facility (TSAF). It includes information for the system administrator to run the TSAF virtual machine, and the service tools provided with it. For the application programmer, it includes a complete description of the Advanced Program-to-Program Communication (APPC/VM) protocol, new with TSAF, and the system services associated with TSAF.

More Information

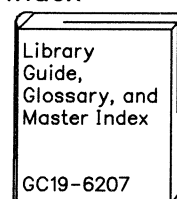
Bibliography

The VM/SP Library (Part 1 of 3)

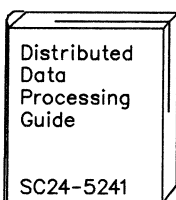
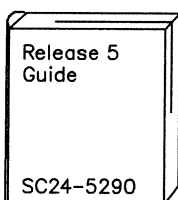
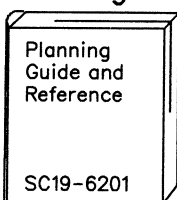
Evaluation



Index



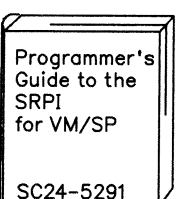
Planning



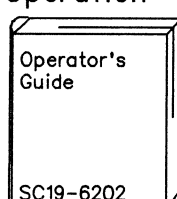
Installation



Applications

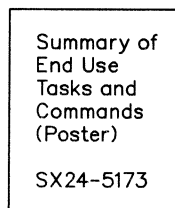
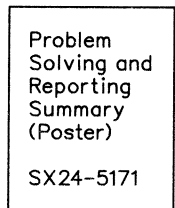
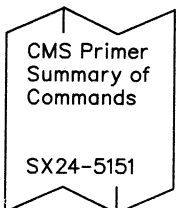
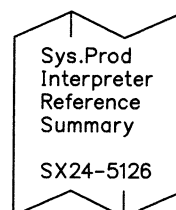
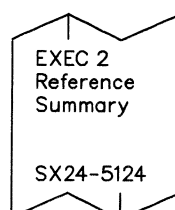
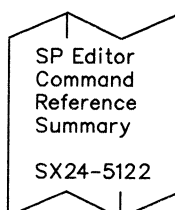
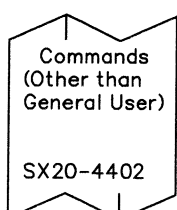
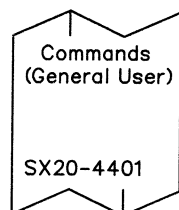


Operation



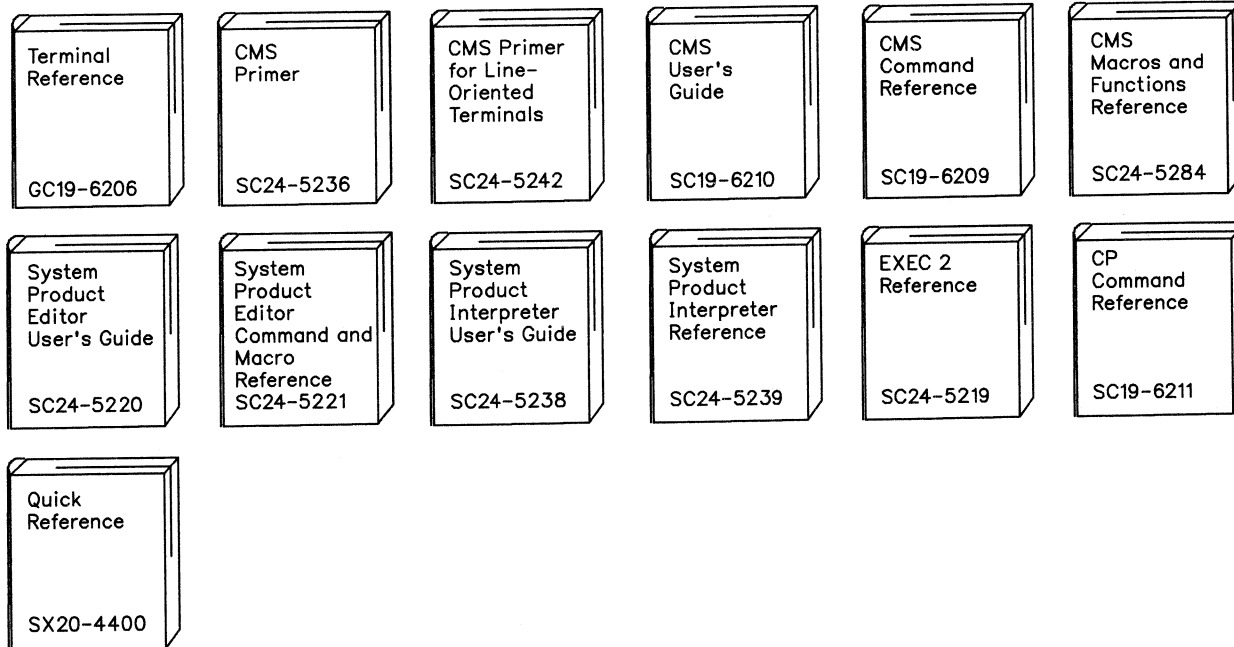
Reference Summaries

To order all of the Reference Summaries, use order number SBOF-3242

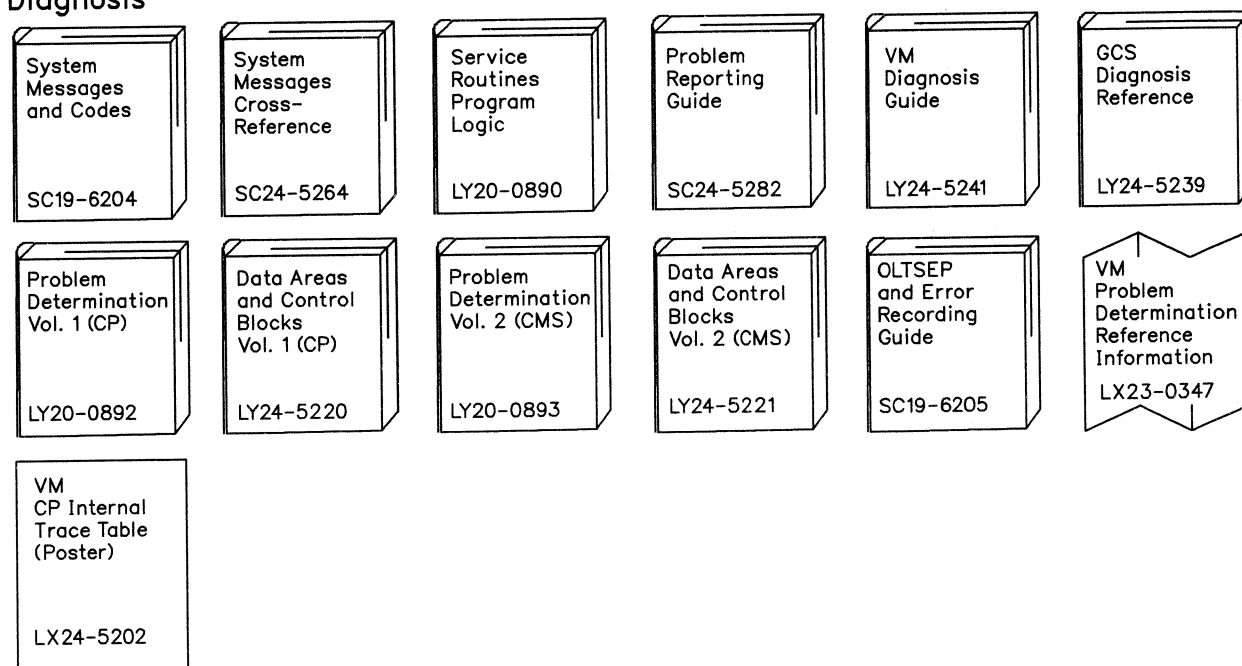


The VM/SP Library (Part 2 of 3)

End Use

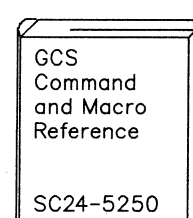
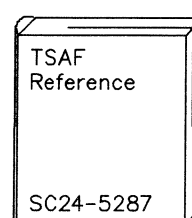
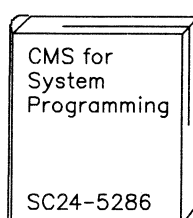
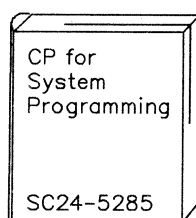
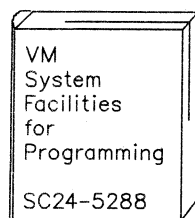


Diagnosis

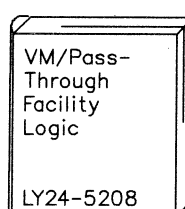
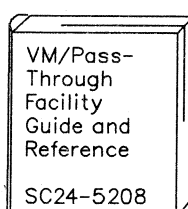
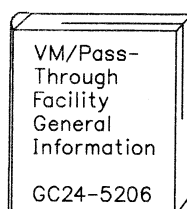
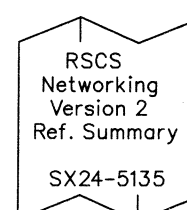
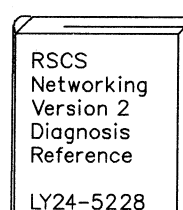
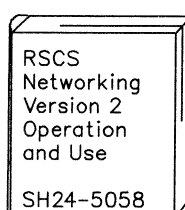
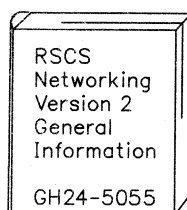
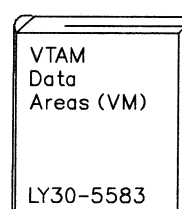
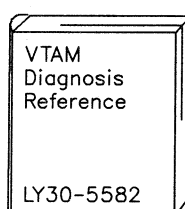
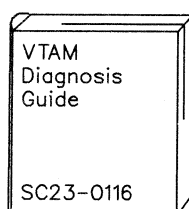
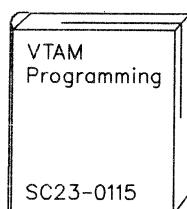
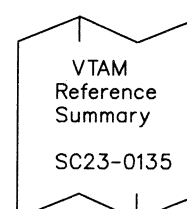
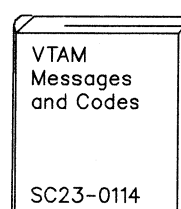
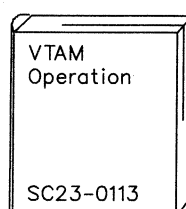
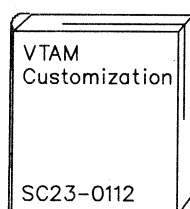
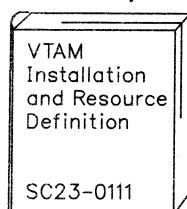


The VM/SP Library (Part 3 of 3)

Administration



Auxiliary Communication Support



A

- abstracts of VM/SP manuals
 - Application Development Guide 267
 - CMS Command Reference 267
 - CMS Primer 267
 - CMS Primer for Line-Oriented Terminals 267
 - CMS User's Guide 267
 - CP Command Reference for General Users 268
 - Distributed Data Processing Guide 268
 - EXEC 2 Reference 268
 - General Information 269
 - Group Control System Macro Reference 269
 - Installation Guide 269
 - Macros and Functions 269
 - Operator's Guide 269
 - Planning Guide and Reference 270
 - Release 5 Guide 270
 - RSCS Networking Version 2 General Information 270
 - Running Guest Operating Systems 270
 - Server-Requester Programming 269
 - System Facilities for Programming 268
 - System Messages and Codes 270
 - System Messages Cross-Reference 270
 - System Product Editor Command and Macro Reference 271
 - System Product Editor User's Guide 271
 - System Product Interpreter Reference 271
 - System Product Interpreter User's Guide 271
 - System Programming 267, 268
 - Terminal Reference 271
 - TSAF Reference 271
- ACCESS command
 - batch job statement 62
 - example of use 25
- ACF/NCP (Advanced Communications Function/Network Control Program) 139
- ACF/SSP (Advanced Communications Function/System Support Programs) 139
- ACF/VTAM (Advanced Communications Function for VTAM) 135, 138
- address space 7
- address translation 20
- ADSTOP command, debugging aid 53
- advanced application, COBOL 197
- Advanced Communications Function for VTAM (ACF/VTAM) 135, 138
- Advanced Communications Function/Network Control Program (ACF/NCP) 139
- Advanced Communications Function/System Support Programs (ACF/SSP) 139
- Advanced Program-to-Program Communication 131
- AFFINITY option for system generation 92
- aid
 - PL/I debugging 200
 - program development, COBOL 196
- allocation of processor time 17
- APL, VS 211
- APPC/VM 131
- APPCVM macro 133
- application
 - advanced, COBOL 197
 - checking new programs 51
 - debugging applications 50
 - developing under CMS 44
 - executing under CMS 57
 - facility for debugging 51
 - IBM Enhanced Connectivity Facilities 238
 - program products 163, 193
 - steps in developing 44
- Application System (AS) 165
- apply local updates to VM/SP 261
- ASMGEND EXEC
- ASSEMBLE command
 - definition 59
 - example of use 45
- assemble, defined 44
- ASSEMBLER H 203
- assembler language 203
- ASSEMBLER XF 45
- assist, virtual machine
 - See virtual machines, assist
- ATTACH command 14
- attribute, LRECL 115
- authorization to use system 88
- AUTHORIZE command, defined 40
- automated CMS initialization 99
- automated procedures 103
- AUTOSAVE option, example of use 112
- auxiliary processor, VS APL 212

B

- BACKSPAC command 30
- BASIC, IBM 214
- batch job statement
 - ACCESS 62
 - LINK 62

- batch processing under CMS 61
- batch processing, VM Batch Subsystem 63
- Batch Subsystem
 - See VM Batch Subsystem
- buffer, terminal input 124
- business professional product
 - SQL/DS with QMF 189
 - VS APL 211

C

- central processing unit 11
- central site, DDP defined 221
- CHANGE command 30
 - batch machine output 63
 - example of use 27
- class
 - device 85
 - privilege 74
- CLASS control statement 74, 77
- CLASS OVERRIDE control statement 74
- CLOSE command, batch machine output 63
- CMS (Conversational Monitor System)
 - assembling programs 45
 - batch processing 61
 - communicating with 99
 - compiling programs 45
 - definition 5
 - developing applications 44
 - dump 55
 - EXEC procedure 107
 - executing programs 57
 - facility for debugging programs 51
 - file system 113
 - general description 97-126
 - HELP facility 121
 - initializing 99
 - introductory description 5
 - message 34
 - minidisk 23
 - subcommands 102
 - user task 5, 105
 - user's view 104
- CMSBATCH command, example of use 61
- CMSEND EXEC
- COBOL 195
- Collection Resource Management System
 - Service 134
- color graphics 182
 - GDDM 182
- command language processor 107
- commands
 - ACCESS 25
 - ADSTOP 53
 - ASSEMBLE 45
 - ATTACH 14
 - AUTHORIZE 40

- BACKSPAC 30
- CHANGE 27
- CLOSE 63
- CMSBATCH 61
- COPYFILE 122
- CP 39
- DEBUG 51
- DEFINE 25, 78
- DETACH 25, 78
- DIAL 151
- DIRMAIN 149
- DISCARD 28
- DISCONN 78
- DISPLAY 53
- DRAIN 31
- DTRIPF 144
- EXECIO 125
- FILEDEF 51, 57, 59
- FLUSH 31
- FORCE 39
- FREE 31
- general 39
- GENMOD 59
- GET LOG 232
- GLOBAL 51
- HASM 204
- HOLD 31
- INCLUDE 59
- INDICATE 15
- LINK 25, 78
- LOAD 59
- LOADBUF 31
- MIGRATE 22
- MONITOR 91
- NAMES 43
- NCCF PROP 233
- ORDER 30, 35
- PASSTHRU 151
- PEEK 28
- PER 53
- PLIOPT 59
- privileged 39
- PROP 232
- PURGE 30, 42
- QUERY 15
- RDRLIST 28, 43
- RECEIVE 28, 42, 102
- REPEAT 31
- RUN 59
- SCRIPT 174
- SENDFILE 30, 43
- SET 35, 40, 78, 108
- SET MSG 40
- SHUTDOWN 35, 39
- SPACE 31
- SPMODE 93
- SPOOL 27, 28, 29, 31
- SPTAPE 31
- START 59

- SVCTRACE 52
- TELL 233
- TERMINAL 79
- TESTFORT 210
- TRACE 52
- TRANSFER 30
- VARY 39
- VMDUMP 55
- XEDIT 101
- communicating
 - between virtual machines 83
 - between VM/SP and work stations 127, 235
 - system operator with VM/SP 35
 - user with VM/SP 34
 - with CMS 99
 - with CP 39, 73
 - with other virtual machines 40
 - with system operator 41
- compile option, TEST 210
- compile, defined 44
- compilers
 - COBOL 196
 - documentation references 48
 - PL/I 199
 - supported by VM/SP 45
- component, network 226
- concepts of VM/SP 36
- configuration, defining device 85
- configurations for VM/IS 249
- connecting
 - work stations to VM/SP 127, 235
- Connectivity Facilities on VM/SP
 - applications 238
 - description 235
 - for VM/SP 127
 - servers and requesters 127
 - SRPI 127, 239
 - starting 127
 - writing applications 239
- console 10
 - definition 10
 - disconnected 79
 - single console image facility 79
 - spooling 31, 63
 - stack 124
 - virtual console management 79
- Console Communication Service (CCS) 84
- CONSOLE directory control statement 77, 79
- console spooling 31
- console usage by system operator 10
- Control Program (CP)
 - See also CP (Control Program)
 - command 39
 - communicating with 39
 - definition 3
 - facilities for debugging programs 52
 - general description 71-96
 - introductory description 3
 - message 34
 - processor time allocation 17
- real resource management 16
- real storage allocation 17
- running operating systems 65
- spool management 19, 26
- system service 84
 - Console Communication 84
 - DASD Block I/O 84
 - Message 84
 - Signal System Service 84
- virtual machine creation 16
- virtual storage management 19
- control statements
 - CLASS 74, 77
 - CLASS OVERRIDE 74
 - CONSOLE 77
 - DEDICATE 74
 - MDISK 77
 - USER 77
- control word, SCRIPT 172
- Conversational Monitor System
 - See CMS (Conversational Monitor System)
- Conversational Monitor System (CMS)
 - assembling programs 45
 - communicating with 99
 - compiling programs 45
 - definition 5
 - developing applications 44
 - dump 55
 - EXEC procedure 107
 - executing programs 57
 - facility for debugging programs 51
 - file system 113
 - general description 97-126
 - HELP facility 121
 - initializing 99
 - introductory description 5
 - message 34
 - minidisk 23
 - subcommands 102
 - user task 5, 105
 - user's view 104
- COPYFILE command, example of use 122
- CP (Control Program)
 - command 39
 - communicating with 39
 - definition 3
 - facilities for debugging programs 52
 - general description 71-96
 - introductory description 3
 - message 34
 - processor time allocation 17
 - real resource management 16, 73
 - real storage allocation 17
 - running operating systems 65
 - spool management 19, 26
 - system service 84
 - Console Communication 84
 - DASD Block I/O 84
 - Message 84
 - Signal System Service 84

virtual machine creation 16
virtual storage management 19
cpu
See processor

D

DASD (direct access storage device)
See also direct access storage device (DASD)
factors affecting requirements 257
minidisk 10, 14, 24
preferred DASD area 22
shared DASD 14
specified in system directory 23
virtual storage 8
DASD Block I/O System Service 84
data base (SQL) 154
data file, sharing 14
data integrity 87
DCF (Document Composition Facility) 171
DDP (distributed data processing)
See also distributed data processing (DDP)
advantages 225
definition 221
distributed site requirements 228
network components 226
programmable operator facility 230
related terms defined 221
VM/SP facilities 225
DEBUG command described 51
debugging
ADSTOP command 53
aid, PL/I 200
application programs 50
VM/SP 95
DEDICATE control statement 74
dedicated tape device 14
DEFINE command, example of use 25, 78
defining system device configuration 85
DETACH command 25
example of use 78
developing applications 44
device, unit record 9
devices
class 85
configuration, defining 85
VM/SP input and output 85
DIAGNOSE X'94' 55
diagnosing system programming problems 95
DIAL command 151
direct access storage device (DASD)
factors affecting requirements 257
minidisk 10, 14, 24
preferred DASD area 22
shared DASD 14
specified in system directory 23
virtual storage 8

Directory Maintenance 147
directory, system
data contained 16
definition 10
for RSCS systems 142
maintenance 147
Pass-Through directory 151
virtual machine definition 16, 74, 77
DIRMAIN command, uses 149
DISCARD command 28
DISCONN command, example of use 78
disconnected console 79
DISPLAY command, as debugging aid 53
Display Management System for CMS
(DMS/CMS) 217
DisplayWrite/370 175
distributed data processing (DDP)
advantages 225
definition 221
distributed site requirements 228
network components 226
programmable operator facility 230
related terms defined 221
VM/SP facilities 225
distributed site requirements 228
distributed site, DDP defined 221
distributing and receiving documents 169
DMKRIO file, defined 255
DMKSNT file, defined 256
DMKSYS file, defined 256
DMS/CMS (Display Management System for
CMS) 217
Document Composition Facility (DCF) 171
document preparation 166
document preparation (PROFS) 169
document, proofreading 169
DOSGEN EXEC
DRAIN command 31
DTRIPF command, example of use 144
dump data used by IPCS 55
dynamic address translation 20

E

editing, full-screen 110
editor, image symbol 181
editor, VM/SP 110
Enhanced Connectivity Facilities on VM/SP
applications 238
description 235
for VM/SP 127
servers and requesters 127
SRPI 127, 239
starting 127
writing applications 239
EXEC procedure described 107
EXECIO command options 125

EXECUTE facility of SQL/DS 188
executing programs under CMS 57
extended control-program support option, SET
command 93

F

FAVORed option of SET command 92
feedback file of the programmable operator
facility 232
file identifier
filemode, definition and use 115
filename, definition and use 115
filetype, definition and use 115
file sharing 116
File Storage Facility program offering 116
FILE subcommand of XEDIT, example of use 112
file system of CMS 113
file transfer 239
FILEDEF command
definition 59
example of use 51, 57
filemode, definition and use 115
filename, definition and use 115
filetype, definition and use 115
fixed-length record, description 113
fixing system problems 95
FLUSH command 31
FORCE command 39
forecasting 165
formatting and printing documents 169
formatting minidisks 25
FORTRAN 206
FORTRAN Interactive Debug 209
frame 20
FREE command, operator use 31
full-screen editing 110
full-screen mode, HELP facility 121

G

GCS (Group Control System)
commands 136
dump 55
installing 259
introductory description 6, 135
profile 88, 136
starting a session 136
virtual machine group 136
GDDM (Graphical Data Display Manager) 180
general command 39
general user
communicate with VM/SP 34
console usage 10
view of spooling 29

Generalized Markup Language (GML) 173
GENMOD command, definition 59
GET LOG command, programmable operator
facility 232
GLOBAL command
definition 59
example of use 51, 58
GML (Generalized Markup Language) 173
Graphical Data Display Manager (GDDM) 180
graphics 166
graphics, color 182
GDDM 182
Group Control System (GCS)
See also GCS (Group Control System)
commands 136
dump 55
installing 259
introductory description 6, 135
profile 88, 136
starting a session 136
virtual machine group 136
group, virtual machine 136
guest operating system 13

H

HASM command to invoke assembler H 204
HELP facility
description 121
example 102
High Performance Option (HPO)
See also HPO (High Performance Option)
free storage management improvements 158
page migration enhancement 158
single processor mode 93
308x processor complex 158
3090 processor complex 158
HOLD command, operator use 31
host data base access 239
HPO (High Performance Option) 157
free storage management improvements 158
page migration enhancement 158
single processor mode 93
308x processor complex 158
3090 processor complex 158

I

IBM BASIC 214
identifier, file
Identify System Service 134
image symbol editor for GDDM 181
INCLUDE command, defined 59
INDICATE command, defined 15

- initializing CMS 99
- initiating Pass-Through sessions 151
- input buffer, terminal 124
- input device 85
- installing VM/IS 249
- installing VM/SP
 - starter system 258
 - using existing system 259
- instruction, macro 204
- Integrated System (VM/IS) 247
- integrity
 - data 87
 - data, during editor session 112
 - power failure 89
 - SQL data base 156
- Inter-User Communications Vehicle (IUCV)
 - communication between virtual machines 83
 - communication with CP system services 84
 - description 82
- interactive menus 217
- Interactive Problem Control System (IPCS) 54
- Interactive Productivity Facility (IPF) 144
- Interactive SQL (ISQL) 156
- Interactive System Productivity Facility (ISPF) 144
- interactive, defined 3
- invoking assembler H 204
- IPCS (Interactive Problem Control System) 54
- IPF (Interactive Productivity Facility) 144
- ISPF (Interactive System Productivity Facility) 144
- ISQL (Interactive SQL) 156
- IUCV (Inter-User Communications Vehicle)
 - communication between virtual machines 83
 - communication with CP system services 84
 - description 82

K

keys for program function (PF) 111

L

- language processors 107
- libraries
 - COBOL subroutine 197
 - FORTTRAN subroutine 206
 - PL/I subroutine 201
- licensed program
 - COBOL 195
 - defined 46
 - Directory Maintenance 147
 - Display Management System (DMS/CMS) 217
 - Document Composition Facility (DCF) 171
 - FORTTRAN 206

- FORTTRAN Interactive Debug 209
- Graphical Data Display Manager (GDDM) 180
- High Performance Option (HPO) 157
- IBM BASIC 214
- Interactive Problem Control System (IPCS) 54
- Interactive Productivity Facility (IPF) 144
- Pass-Through Facility 150
- PL/I 199
- Professional Office System (PROFS) 168
- Remote Spooling Communications Subsystem (RSCS) Networking 141
- service provided 6
- SQL/Data System 154, 185
- SQL/Data System with QMF 189
- VS APL 211
- LINK batch job statement 62
- LINK command, example of use 25, 78
- link, DDP defined 221
- LISTING file 45, 46, 51
- LOAD command
 - defined 59
 - example of use 51
- LOADBUF command 31
- Logging on to VM/SP
- LRECL attribute, example of use 115

M

- macro instruction, definition and example 204
- magnetic tape device 14
- main storage
 - See real resource
- manual, VM/SP
 - See abstracts of VM/SP manuals
- mapping virtual to real storage 9
- maximum data items on minidisk 115
- maximum storage size 7, 19
- MDISK control statement 77
- memos 166
- menus for interactive environment 217
- Message System Service 84
- messages
 - sending to system operator 41
 - system 34
- MIGRATE command, example of use 22
- migration process 65
- minidisks
 - access to 25
 - concept 10
 - description 24
 - how defined 23
 - initializing 25
 - maximum data items 115
 - sharing 10
- MODULE file 57
- MONITOR command, defined 91
- multiple access, defined 3

multiple shadow table option, SET command 92

N

NAMES command, example of use 43
NCCF (Network Communications Control Facility) 230, 232
NCCF PROP command 233
NCCF PROP command, programmable operator 233
Network Communications Control Facility (NCCF) 230, 232
network component 226
network, DDP defined 221
node
 central site requirement 227
 defined 221
 Pass-Through 151

O

object program, defined 44
office task program product
 DCF 171
 GDDM 180
 PROFS 170
operating system
 executing 65
 guest under VM/SP 13
 shared DASD 14
optimizing compiler, PL/I 199
option, performance 90
ORDER command
 batch machine output 63
 example of use 30, 35
output device 85

P

page
 frame 20
 migration 22
 of storage 20
paging
 description 21
 overview 20
 preferred DASD area 22
panel
 HELP 122
 interactive 217
Pass-Through Facility 150
PASSTHRU command 151

password, readshare 87
PCF (Problem Control Facility) 145
PEEK command, example of use 28
PER command, description 53/
performance option 90
PF (program function) key 111
PL/I
 debugging aid 200
 description 199
 optimizing compiler example 46
planning for VM/SP
 device configuration 85
 preparing system files 255
 storage requirement 256
planning model 165
PLIOPT command, defined 59
power failure, data protection 89
POWERINP description 110
preferred DASD area 22
PREPARE facility of SQL/DS 188
preparing system files 255
preprocessor for SQL/DS 185
printers
 spooled virtual 27
 virtual 42
priority
 operator's virtual machine 35
 spooled file 27
 virtual machine 17, 31
privilege classes 74
privileged command 39
Problem Control Facility (PCF) 145
problem diagnosis 95
problem reporting 54
procedure, automated 103
processing batch jobs
 under CMS 61
 VM Batch Subsystem 63
processing unit, central 11
processor
 command language 107
 sharing usage 11
 time allocation 17
Professional Office System (PROFS) 168
PROFILE EXEC 79, 99
PROFILE GCS 88, 136
PROFS (Professional Office System) 168
program function (PF) key 111
program, licensed
 See licensed program
programmable operator facility 230
 command subset 233
 feedback file 232
 how it works 231
 NCCF PROP command 233
 PROP command 232
 PROPST EXEC 232
 routing table 231
 tasks performed 230
 user's view 232

- programming problem diagnosis 95
- programs
 - development aid, COBOL 196
 - object 44
 - product 163, 193
 - source 44
 - stack 124
 - update tape (PUT) 261
- project control 166
- proofreading documents using PROFS 169
- PROP command 232
- PROPST EXEC 232
- publications for VM/SP
 - See abstracts of VM/SP manuals
- punch, virtual 41
- PURGE command
 - batch machine output 63
 - example of use 30, 42
- PUT (program update tape) 261
- PVM dump 55

Q

- QBE (Query By Example) 190
- QMF (Query Management Facility) 189
- Query By Example (QBE) 190
- QUERY command
 - defined 15
 - example of use 35, 42, 78, 102
- Query Management Facility (QMF) 189
- queue drop elimination option of SET command 93

R

- RDRLIST command, example of use 28, 43
- reader, virtual
 - defined 41
 - handling input files 42
- readshare password 87
- real resource
 - managed by CP 3
 - real console 10
 - real storage 9
 - sharing 5
- real storage
 - allocation 17
 - mapping to virtual 9
 - security 88
- RECEIVE command, example of use 28, 42, 102
- RECFM attribute, example of use 115
- record, data 113
- record, variable-length 114
- remote site, DDP defined 221
- Remote Spooling Communications Subsystem (RSCS) networking 67, 141

- REPEAT command 31
- reporting, problem 54
- request, status information 15
- requesters
 - description 127, 239
- reserved page option, SET command 92
- resource management, real 16, 73
- Restructured Extended Executor (REXX)
 - language 109
- retrieving documents, PROFS 170
- REXX (Restructured Extended Executor)
 - language 109
- routing table for programmable operator 231
- RSCS (Remote Spooling Communications Subsystem) networking 141
- RSCS dump 55
- RUN command 59
- RUNTSAF command 133

S

- SAVE subcommand of XEDIT, example of use 112
- screen, DMS/CMS 218
- SCRIPT command 172
 - control word 172
- SCRIPT/VS (DCF) 172
- security
 - data 87
 - real storage 88
- segment 20
- SENDFILE command, example of use 30, 43
- sending messages to system operator 41
- Server-Requester Programming Interface (SRPI)
 - See SRPI (Server-Requester Programming Interface)
- servers
 - description 127, 239
- servicing VM/SP
 - program update tapes (PUT) 261
- session manager, VS APL 212
- SET command
 - example of use 35, 40, 78, 108
 - extended control-program support option 93
 - favorable execution option 92
 - multiple shadow table option 92
 - queue drop elimination option 93
 - reserved page option 92
 - user priority option 92
 - virtual machine assist option 93
- SET SMSG command 40
- SET subcommand of XEDIT, definition 111
- shared DASD 14
- shared segment 88
- sharing data files 14, 116
- sharing system resources 7
 - data integrity 87
 - direct access storage 14

- minidisk 10, 25
- processor usage 11, 17
- real storage segment 88
- through spooling 26
- user CMS file 87
- SHUTDOWN command, example of use 35, 39
- Signal System Service 84
- single console image facility 79
- single processor mode
 - option, SPMODE command 93
- slicing, time 11
- SNA (Systems Network Architecture)
 - benefits 135
 - defined 135
 - introductory description 6, 135
 - network components 227, 228
 - node 142
- SORT subcommand of XEDIT, example of use 111
- source program, defined 44
- SPACE command 31
- SPMODE command, example of use 93
- SPOOL
 - batch job statement 62
 - batch machine output 63
 - command 27, 42
 - command, example of use 28, 29, 31
 - control statement 77
- SPOOL CONSOLE command 30
- spool operator task 30
- spooled virtual printer 27
- spooling 26
 - CP management 19, 30
 - description 26
 - example 26
 - overview 26
 - priority 27
 - resulting from I/O requests 29
 - summary of CP activity 32
 - system operator task 30
 - user control 30
 - virtual console data 31
 - virtual punch 28
 - virtual reader 28
- SPTAPE command 31
- SQL (Structured Data Language)/Data System 185
- SQL (Structured Query Language)/Data System 68
 - data base 156
 - EXECUTE facility 188
 - PREPARE facility 188
 - preprocessor 185
 - with QMF 189
- SRPI (Server-Requester Programming Interface)
 - description 127, 239
- STACK subcommand of XEDIT, example of use 111
- stack, console 124
- stack, program 124
- stand-alone dump 55
- START command
 - defined 59
 - example of use 51
 - operator use 31
- starter system 258
- statistics 165
- status information request 15
- storage
 - address translation 20
 - allocation 17
 - how CP manages 17
 - how organized 20
 - maximum size 7
 - planning for requirements 256
 - virtual address 20
 - virtual storage 7
 - virtual vs. real 9
- STORE command 53
- Structured Query Language (SQL)/Data System
 - See SQL (Structured Query Language/Data System)
- Structured Query Language/Data System (SQL) 68
 - data base 156
 - EXECUTE facility 188
 - PREPARE facility 188
 - preprocessor 185
 - with QMF 189
- subcommand, CMS 102
- summary of VM/SP concepts 36
- SVCTRACE command described 52
- symbol editor for GDDM 181
- system directory
 - See directory, system
- system file, preparing 255
- system generation option
 - AFFINITY 92
 - virtual = real 92
- system message 34
- system operator
 - batch virtual machine 61
 - communication with VM/SP 35
 - console usage 10
 - controlling performance options 91
 - dedicating devices 14
 - IPF panel 145
 - privileged command 39
 - RSCS Networking management task 143
 - sending messages to 41
 - use of CMS commands 105
 - view of spooling 30
 - VM/Pass-Through management 152
- System Product Editor
 - description 110
 - using for program development 45
- System Product Interpreter 109
- Systems Network Architecture (SNA)
 - See also SNA (Systems Network Architecture)
 - benefits 135
 - defined 135
 - introductory description 6, 135
 - network components 227, 228
 - node 142

T

tag, DCF 173
 tape device, magnetic 14
 task
 CMS user 5, 105
 spool operator 30
 TELL command, example of use 233
 TERMINAL command, example of use 79
 terminal input buffer 124
 terminals 10
 TEST compile option 210
 TESTFORT command 210
 TEXT files 45, 46, 57
 text preparation program product
 DCF 171
 PROFS 168
 text processing 166
 time allocation, processor 17
 time slicing 11, 17
 TRACE command described 52
 TRANSFER command, example of use 30
 Transparent Services Access Facility (TSAF) 131
 TSAF (Transparent Services Access Facility) 131
 TXTLIB (TEXT file library) 58

U

unit record device
 defined 9
 in a virtual machine 14
 virtual device 9
 update
 UPDATE, update procedures 261
 update procedures
 applying local updates 261
 ASMGEND 261
 CMSGEN 261
 DCSSGEN 262
 deciding which procedure to use 261
 DOSGEN 262
 GENTSAF 262
 PRELOAD 262
 SAMGEN 262
 SPGEN 262
 VMFLKED 262
 VMFNLS 262
 VMFREMOV 262
 VMSERV 262
 VRSIZE 262
 VSAMGEN 262
 VSEVSAM 262
 ZAP 262
 ZAPTEXT 262
 user class 74

USER control statement 77
 user priority option for SET command 92
 user task, CMS 5, 105
 user view, CMS 104
 user, general
 See general user

V

variable-length records 114
 VARY command, example of use 39
 vector symbol editor (GDDM) 181
 virtual address 20
 virtual concept 7-11
 device 9
 console 10
 direct access storage 10
 magnetic tape 10
 unit record 9
 storage 7
 compared to real 9
 need for 7
 size limitation 7
 summary 36
 virtual console data, spooling 31
 virtual console management 79
 virtual disks 238
 virtual file 238
 Virtual Machine/System Product (VM/SP)
 recommended updating procedures 261
 virtual machines
 assist 93
 option of SET command 93
 batch machine 61
 changing the definition 17
 console 10, 79
 defining in directory 16
 description 12
 display status 39
 group 136
 input and output 41
 make changes 40
 operating system 13
 priority 17
 requirement 13
 resource available to 12
 share of processor time 17
 use of real devices 19
 virtual input/output device 14
 working in virtual environment 39
 virtual print 239
 virtual printer 27, 42
 virtual punch 28
 spooling 28
 virtual reader
 defined 41
 handling input files 42

- spooling 28
- virtual storage 7
 - mapping to real 9
 - virtual storage management 19
- Virtual Telecommunications Access Method (VTAM) 135, 138
- virtual = real system generation option 92
- VM Batch Subsystem 63
- VM File Storage Facility 116
- VM SNA Console Support (VSCS) 139
- VM/HPO
 - See High Performance Option (HPO)
 - See HPO (High Performance Option)
- VM/IS (Integrated System) 247
- VM/Pass-Through 68
- VM/SP (Virtual Machine/System Product)
 - defined 3
 - recommended updating procedures 261
- VM/SP concepts 36
- VM/SP manuals
 - See abstracts of VM/SP manuals
- VMDUMP 55
- VS APL 211

- VSCS 139
- VTAM (Virtual Telecommunications Access Method) 135, 138

X

- XEDIT command, example of use 101, 111
 - FILE subcommand 112
 - POWERINP subcommand 110
 - SAVE subcommand 112
 - SET subcommand 111
 - SORT subcommand 111
 - STACK subcommand 111

Numerics

- 308x processor complex, VM/HPO 158
- 3090 processor complex, VM/HPO 158

**International Business
Machines Corporation
P.O. Box 6
Endicott, New York 13760**

**File No. S370/4300-20
Printed in U.S.A.**

GC19-6200-4



Virtual Machine/System Product
Introduction
Order No. GC19-6200-4

**READER'S
COMMENT
FORM**

Is there anything you especially like or dislike about this book? Feel free to comment on specific errors or omissions, accuracy, organization, or completeness of this book.

If you use this form to comment on the online HELP facility, please copy the top line of the HELP screen.

_____ Help Information line ____ of ____

IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you, and all such information will be considered nonconfidential.

Note: Do not use this form to report system problems or to request copies of publications. Instead, contact your IBM representative or the IBM branch office serving you.

Would you like a reply? ☐ YES ☐ NO

Please print your name, company name, and address:

IBM Branch Office serving you: _____

Thank you for your cooperation. You can either mail this form directly to us or give this form to an IBM representative who will forward it to us.

Reader's Comment Form

CUT
OR
FOLD
ALONG
LINE

Fold and tape

Please Do Not Staple

Fold and tape



BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NY

POSTAGE WILL BE PAID BY ADDRESSEE:



INTERNATIONAL BUSINESS MACHINES CORPORATION
DEPARTMENT G60
PO BOX 6
ENDICOTT NY 13760-9987

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES



Fold and tape

Please Do Not Staple

Fold and tape

