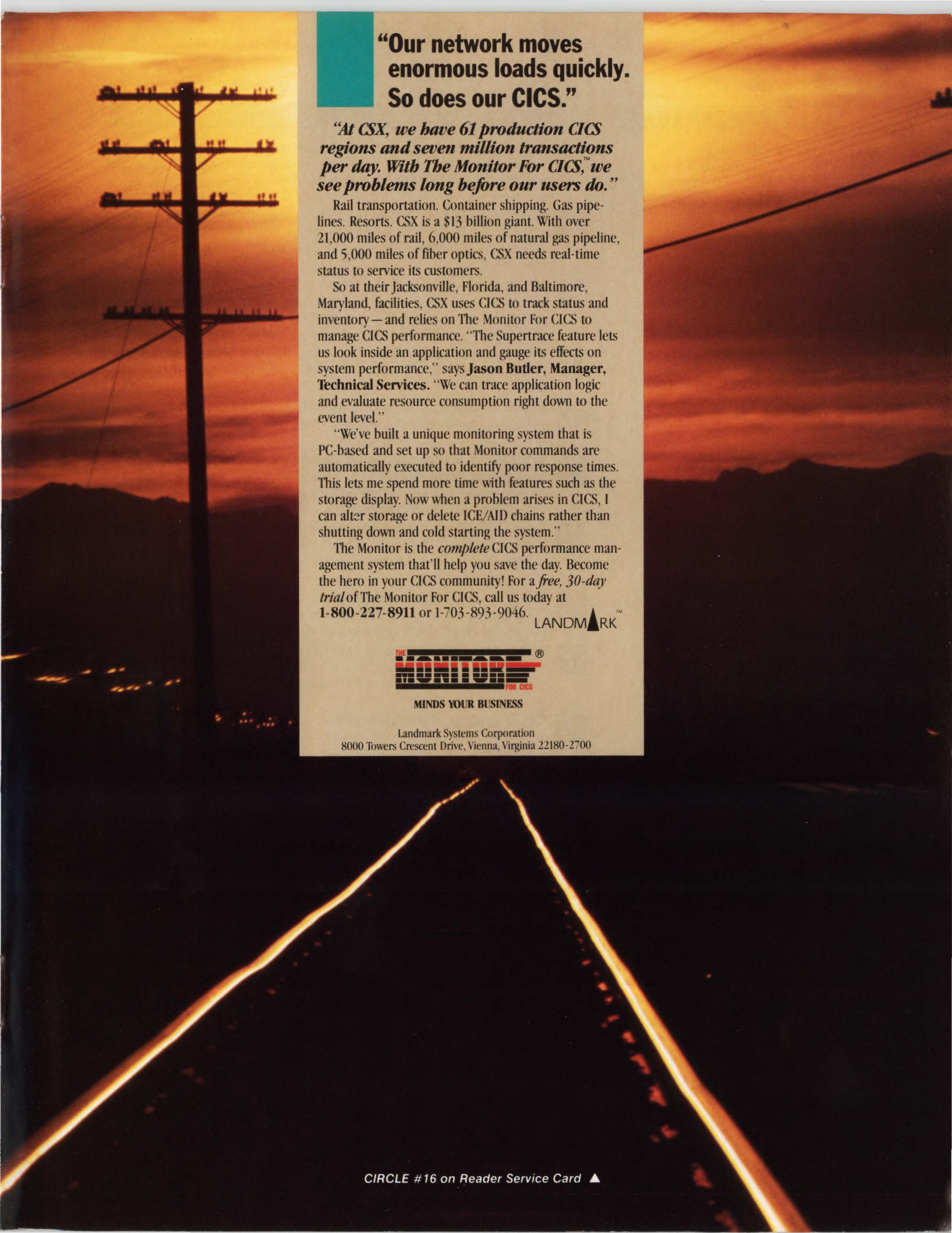
MAINFRAME JOURNAL

For Users of IBM System/370 Architecture & Compatible Systems

March 1989

NOREX Leasing Inc. Reduces Processing Time With VSAM Replacement


"Our network moves enormous loads quickly. So does our CICS."
"At CSX, we have 61 production CICS regions and seven million transactions per day. With The Monitor For CICS,™ we see problems long before our users do."
Rail transportation. Container shipping. Gas pipelines. Resorts. CSX is a $13 billion giant. With over 21,000 miles of rail, 6,000 miles of natural gas pipeline, and 5,000 miles of fiber optics, CSX needs real-time status to service its customers.
So at their Jacksonville, Florida, and Baltimore, Maryland, facilities, CSX uses CICS to track status and inventory—and relies on The Monitor For CICS to manage CICS performance. "The Supertrace feature lets us look inside an application and gauge its effects on system performance," says Jason Butler, Manager, Technical Services. "We can trace application logic and evaluate resource consumption right down to the event level."
"We've built a unique monitoring system that is PC-based and set up so that Monitor commands are automatically executed to identify poor response times. This lets me spend more time with features such as the storage display. Now when a problem arises in CICS, I can alter storage or delete ICE/AID chains rather than shutting down and cold starting the system."
The Monitor is the complete CICS performance management system that'll help you save the day. Become the hero in your CICS community! For a free, 30-day trial of The Monitor For CICS, call us today at 1-800-227-8911 or 1-703-893-9046.
LANDMARK™
THE MONITOR® FOR CICS
MINDS YOUR BUSINESS
Landmark Systems Corporation
8000 Towers Crescent Drive, Vienna, Virginia 22180-2700
CIRCLE #16 on Reader Service Card ▲

CONTENTS

March 1989

Volume IV
Number 3

DEPARTMENTS

FEATURES

COVER:

NOREX Leasing Inc. (Toronto, Ontario), the largest leasing company in Canada, provides financial alternatives for companies to acquire trucks, aircraft, computer systems and other necessary equipment. The company recently reduced its processing time with the help of a VSAM replacement product. See the article on page 31. Photograph by Grant V. Faint.

MAINFRAME JOURNAL© (ISSN 0895-5751) is published monthly by Thomas Publications, Inc., 10935 Estate Lane, Suite 375, Dallas, TX 75238, (214) 343-3717. Second class postage paid at Dallas, TX.
POSTMASTER: Send address changes to: MAINFRAME JOURNAL, P.O. Box 551628, Dallas, TX 75355-1628.

PUBLISHER'S COMMENTS

MAINFRAME CONFERENCE — Is The Time Right?

Bob Thomas

In a typical medium-to-large DP/MIS organization using an IBM mainframe the personnel usually fall into five major segments or categories; DP/MIS Management, Systems/Technical Support, Applications Development/Database Management, Operations, and Communications/Network Management. In an effort to stay abreast of current technology or to explore new alternatives, many of the members of this typical DP/MIS organization fly off in all directions throughout the year to attend conferences, seminars, and classes targeted to their particular job function.

Within the past six months I have been contacted on several occasions by people asking basically the same question, "Would MAINFRAME JOURNAL consider sponsoring or coordinating an IBM mainframe users conference that would be strictly oriented to providing informative technical sessions?" These requestors went on to suggest that the sessions could be given by some of the same people who have written articles for MAINFRAME JOURNAL as well as other knowledgeable individuals.

The intent of such a conference would be to give you the same in-depth type of information you have come to expect from MAINFRAME JOURNAL except that the sessions would provide more information (and more examples) and attendees would have the opportunity to ask questions and receive answers interactively. In addition the sessions would be specifically targeted to each of the five major segments of a DP organization mentioned above. Another particularly appealing facet of such a MAINFRAME CONFERENCE would be a separate vendor exhibit area where only IBM mainframe vendors would be located so that you could see the latest and greatest solutions to today's computing needs.

As I see it, one of the most important factors to the success of such a MAINFRAME CONFERENCE is that each session would be targeted to a specific audience. For example, if topics such as *"How to Benefit From Monitoring CICS", "VSAM Optimization", "Unattended Operations", "Security Issues"* or *"Improving DB2 Performance"* were provided, there would be up to five different sessions on each topic. The difference would be that each session would discuss the topic with an orientation toward the attending audience (e.g. Systems/Technical Support personnel would attend the sessions oriented toward them while DP/MIS Managers would attend sessions on the same topic but oriented toward their interests).

A MAINFRAME CONFERENCE oriented specifically to IBM mainframe users would seem to benefit attendees twofold. First, "under one roof", attendees could select from a wide variety of sessions specifically oriented toward their position or function within their company. Second, also "under one roof", attendees would have the opportunity to examine vendor solutions (software and hardware) featuring only those vendors with IBM mainframe-compatible products. Of course the products offered would span the interest of all five major audience groups.

From my perspective, the most appealing aspect of such a MAINFRAME CONFERENCE is the tremendous amount of "cross-pollination" that would be possible by targeting the sessions and the vendor exhibits to each of the five audience groups as opposed to a conference held with one set of sessions oriented only toward one specific audience within the DP organization. After all, where else could personnel from all five major segments of a DP organization attend conferences and see product demonstrations as diverse as:

- MVS, VSE, & VM Utilities
- Decision Support Systems
- CASE Tools
- DASD Management Software
- Integrated Accounting Packages
- CICS & VSAM Performance Tools
- Security & Disaster Recovery Software
- Unattended Operations Software
- Performance Evaluation Software
- DB2 & IMS Tools
- DBMS Alternatives
- Network Performance Monitors
- Change Control Tools
- Electronic Mail & FAX Systems
- PC Connectivity Tools
- Report Distribution Software
- Programming & Debugging Aids
- and many, many more . . .

MAINFRAME CONFERENCE — Is The Time Right?

Will there be a MAINFRAME CONFERENCE? That is really up to you. If there is sufficient interest in the IBM mainframe-user community then planning for the first MAINFRAME CONFERENCE will begin immediately.

Please take a minute right now while the subject is fresh on your mind to let me know what your thoughts are regarding a MAINFRAME CONFERENCE. Turn to a Reader Service Card following page 58 and jot down your thoughts (or just Circle #226 if you are interested). Your opinion counts. Thanks for taking the time!

MAINFRAME JOURNAL
For Users of IBM System 370 Architecture & Compatible Systems

Publisher and Editor-in-Chief
Bob Thomas

Associate Publisher
Martha Thomas

Editor
Carol M. Hoag

Copy Editors
Judy Beller
Pat Warner

Art Director
David Kramer

Assistant Art Director
Ken Buerer

Advertising Manager
Denise Thomas

Marketing Services
Sally Webb

Circulation Manager
Janice Porter

Assistant Circulation Manager
Nancy Crawford

Administrative Manager
Marian Davenport

Typography Coordinator
Dub Pope
Jaggars Chiles Stovall

Printing Coordinator
Steve Priester
American Signature Graphics

BPA

SUBSCRIPTION RATES & INQUIRIES: Subscriptions are free within the USA and Canada. One-year foreign subscriptions are $96.

All subscriptions, remittances, requests and changes of address should be sent to MAINFRAME JOURNAL at 10935 Estate Lane, Suite 375, Dallas, TX 75238, (214) 343-3717.

READER FORUM

PR/SM VS. MDF

In the article, "MVS Performance Management Legends," by Stephen Samson (November/December 1988), he correctly recognizes the need for an appropriate solution to the problem of operating multiple system images on a single hardware configuration. However, he seems to have forgotten that Amdahl has provided that solution with the Multiple Domain Feature (MDF) in 1985. The February 1988 PR/SM announcement he refers to was simply a response by IBM to the role MDF was filling in the computing industry.

While the author may feel PR/SM "sounds similar" to MDF, if he listens again he will find that the three-year head start MDF has had over PR/SM enables it to offer a multitude of features missing in PR/SM. Some of the specific features are:

- Minimal overhead for multiple domains sharing a single CPU
- The ability to run VM/XA SP R2 in a production environment
- The ability to set minimum, maximum and target values for CPU usage
- The ability to assure service predictability for more than the highest priority task
- One IOCDS for each domain and a domain's IOCDS can be changed with the system running
- The ability to reconfigure memory between domains
- The ability to dynamically partition or join a configuration with the system running
- Command list files to automate operator instructions.

There is no doubt that MDF, in contrast to PR/SM, clearly requires less overhead and gives the customer tighter control of CPU allocation, more flexible configurability and more flexible operations.

David L. Anderson
VP Processor Product Management
Amdahl Corp.
Sunnyvale, CA

KUDOS

I particularly enjoyed Stephen Samson's article, "MVS Performance Management Legends" (November/December 1988), Joel Goldstein's article on DB2 and Smith and Spund's article on SQL/DS.

Gary M. Schultz
Systems & Data Processing
State of Wisconsin
Madison, WI

I found this issue's (January 1989) articles, "Streamlining BMS Data Flow" (by Paul Henken) and "VSAM Optimization And Design" (by Eugene Hudders), both interesting and informative. MAINFRAME JOURNAL is the first magazine geared toward what interests mainframe application developers and in my 16 years in data processing I've read many publications. Please enter my subscription immediately.

Bob Kufert
Applications Devel.
Southeast Bank
Miami, FL

"Managing Data Processing Personalities" (by Ted Keller) in the January 1989 issue was a really great article!

Mark A. Knoell
Director of DP
Empire State News Corp.
Cheektowaga, NY

Since we are a relatively new MVS shop on a shoestring budget, I especially enjoyed the clear and matter-of-fact article, "Poor Man's DASD Management System" (by Wayne Meriwether), in the January 1989 issue. Our IBM reps want us to go ESA and it helps to have sources for input.

Gary R. Simmons
Tech Support
Lee Memorial Hospital
Ft. Meyers, FL

MAINFRAME JOURNAL continues to be the best trade magazine in the industry! It is as if you read my mind. If I hit a gray area, you feature it on the next cover (ESA). Keep up the good work.

John Gargis
Tech Support
Pansophic Systems
Roswell, GA

ISPF

"ISPF Spells Productivity" (by Jon Pearkins) in the November/December 1988 issue was an excellent article to pass on to my junior operators to help them gain insight to ISPF.

Charles P. Mallahan, Jr.
Lead Operator
DST System, Inc.
Kansas City, MO

As a new MVS shop, the article, "ISPF Spells Productivity," will be valuable to our application programmers.

Bob Chambers
Systems Programmer
Union National Bank
Little Rock, AR

The article, "ISPF Spells Productivity," covers many good points but the bias against XEDIT was blatant.

Bill Allen
Systems Programmer
Fruit Of The Loom
Bowling Green, KY

Editor's Note: *You will be happy to know that:*

1.) This issue has another article by Jon Pearkins titled "ISPF Techniques." In addition, Jon will be hosting a series of columns featuring ISPF Tips and Techniques he has collected. If you have any gems you would like to have included send them to: ISPF Tips, MAINFRAME JOURNAL, PO Box 551628, Dallas, TX 75355-1628.

2.) We will begin a series of articles featuring XEDIT starting in April.

GUIDE

MAINFRAME JOURNAL is great! I find the most *useful* information, more than anywhere else. Pete Clark's articles are always terrific. The article on GUIDE built me up but didn't tell how, when or where to attend.

John Wallin
Systems Programmer
Savers Federal S&L Assoc.
Little Rock, AR

Editor's Note: *For more information about GUIDE International, Corp., contact Margaret Miller at GUIDE, 111 E. Wacker Dr., Suite 600, Chicago, IL 60601, (312) 644-6610 Ext. 3216.*

Analyzing VM Performance

A Worksheet-Based Approach

By Dr. Robert Berry and Joseph Hellerstein

The Virtual Machine (VM) operating system has become increasingly popular because of its ease of use and installation as well as its effectiveness for interactive computing and office automation. The recently announced 9370 has assisted in the move toward more departmental and distributed computing. For both software ease of use and hardware requirements, VM is expected to be a principle operating system in this environment.

In comparison to MVS, there is little literature on VM performance management. This article describes a systematic approach to analyzing the VM performance reports produced by VMMAP, the Virtual Machine Monitor Analysis Program. The approach is encapsulated in a worksheet on which performance analysts record performance indicators and describe their conclusions. Over the past year, this worksheet has been used to analyze approximately 30 Virtual Machine/System Product (VM/SP) and Virtual Machine/System Product High Performance Option (VM/SP HPO) systems ranging from 9370 Model 20s to 3090 Model 200s. The worksheet was found to be an effective tool for structuring the analysis of VM performance as well as reporting the results of this analysis.

The VM Performance Analyst

During the past year extensive interviews were conducted with VM performance analysts both inside and outside IBM. The objectives were to identify who requests information from performance analysts, what type of answers they want and, more generally, how performance analysts spend their time. A part of this analysis is summarized in Figure 1.

Since the performance analyst is responsible for collecting and interpreting performance data, he is the focal point for questions about performance. He hears complaints from users or a user council (an organization that participates in the negotiation and enforcement of service level agreements). The problem might be real time (for example, a runaway user), or might have been detected overnight (for example, response for a certain class of users was poor last week). His role is to identify the cause of the problem and respond. To resolve performance problems, an analyst might need information from another department (for example, has operations been experiencing hardware problems?). Once a course of action has been identified, he may not have the authority to implement it. Instead, a request may have to be made to the appropriate group (for example, ask the I/O management group to add a paging area to a specific device). The request may need to be accompanied by a justification.

In short, the performance analyst has many responsibilities; he must interact with many different groups. Further complicating the job is that performance analysts are frequently responsible for more than one system, sometimes more than 10.

VM Data Sources

There are several data sources available in a VM system. Each is appropriate for the investigation of a certain class of problems.

The easiest to use are the CP and CMS commands: INDICATE and QUERY. These commands allow an operator (and to a lesser extent, a user) to interrogate the state of the system. They are useful for determining current settings for certain tuning parameters as well as the values for some global indicators.

For real-time performance problems there are tools that allow snapshots of the system to be taken. Data taken in these snapshots is quite detailed and allows for tracking of specific user or resource behavior. IBM's VM Real Time Monitor and Candle Corporation's (Los Angeles, CA) Omegamon for VM are tools that fit into this category.

Performance tuning usually requires data collected from a different perspective. In general, one is interested in identifying day-to-day trends and chronic problems; minute-to-minute anomalies are not of interest. Throughout the day the VM operating system produces a series of monitor records that can be post processed to identify system bottlenecks. IBM's VMMAP, VMPPF and VMMONITOR from VM Software, Inc. (Reston, VA) are tools that post process VM monitor data. This article focuses on the analysis of data appropriate for performance tuning, specifically the data produced by VMMAP.

VMMAP Analysis

As with any performance reporter, interpreting VMMAP reports is complicated by the following.

Volume of information

Data on every imaginable aspect of system performance is available in a lengthy report. It takes much time to find the information of interest.

Organization of information

Related data is frequently not adjacent in a report.

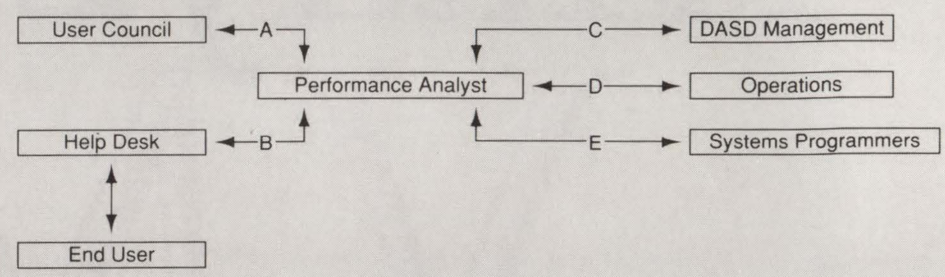
Interpreting the data

Much of the data reported by the VM monitor is of a VM-internals nature. Decoding this information is complicated and requires both general performance analysis skills (such as statistical analysis and analytical modeling) and specific expertise in VM internals.

Also, many people either lack the time or the expertise to study the performance reports.

FIGURE 1

Information Flow For A Performance Analyst

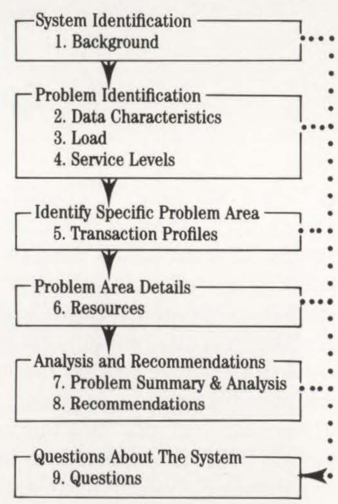


Path	Information
A — to Performance Analyst:	Questions about cause for poor performance. Usually long term.
A — from Performance Analyst:	Reasons, "working on it," "need more info."
B — to Performance Analyst:	Questions about cause for poor performance. Could be real time.
B — from Performance Analyst:	Reasons, "working on it," "need more info."
C — from Performance Analyst:	Requests to change I/O config. or minidisk layout. Questions about config.
C — to Performance Analyst:	"WHY," "done as requested," "done, but our way," "not done."
D — from Performance Analyst:	Questions about system status, Requests to enable/disable equipment.
D — to Performance Analyst:	"WHY," "done as requested," "done, but our way," "not done."
E — from Performance Analyst:	Requests to change system definition or set tuning parameters.
E — to Performance Analyst:	"WHY," "done as requested," "done, but our way," "not done."

A VM Performance Analysis Worksheet

Interviews showed that performance analysts are overwhelmed by the details of interpreting VMMAP data. Often they had the expertise necessary to interpret a problem, but they lacked (or did not remember) a systematic approach for looking at the performance data. To make this process more systematic, a worksheet was introduced that suggests a step-by-step approach to analyzing VM performance data. This methodology was successfully applied to many VM systems, both inside and outside IBM. The worksheet is effective for a wide range of VM systems. To focus the presentation, however, data is used from one system to present the worksheet. This data is taken from an IBM 3081 running the VM/SP HPO 4.42 operating system with a petroleum industry workload.

A high level outline of the worksheet is shown in Figure 2. The organization of the worksheet is complementary to the systematic approach that many performance analysts take to looking at performance data, that is:

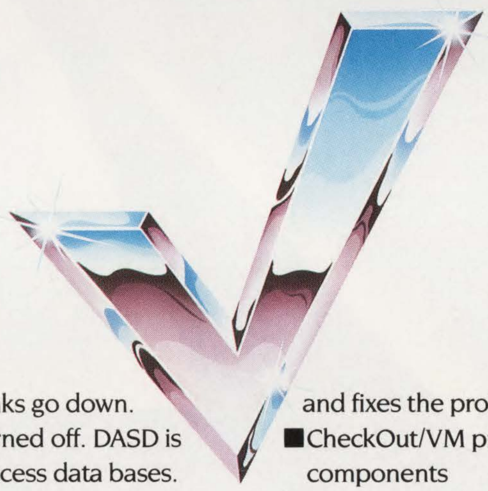


Complete details of the contents of each part of the worksheet are beyond the scope of this article. However, an overview of each section is provided.

Background

The background section includes general information about the system being analyzed. This information is useful for identifying high level system characteristics. For example, the following details are recorded: the VM release level, the hardware it runs on and memory configuration specifics.

CheckOut/VM Has It Up and Running Before You Ever Knew It Was Down



RSCS links drop. Communication links go down. Remote printers are accidentally turned off. DASD is improperly attached. Users can't access data bases.

These are hardware and software component problems which impact VM availability. And down components mean the resources of your VM system are inaccessible to users. When your users can't get to the information they need, you get phone calls. Lots of phone calls. Unpleasant phone calls.

There's a solution. Duquesne Systems announces CheckOut/VM, a new direction in defining and managing availability. It automatically checks the *component* availability of your VM operation to ensure that total system resources are alive and well. Your users experience fewer interruptions due to component downtime. And you get far fewer phone calls.

Just locating a component which has failed can consume a great deal of your time. CheckOut/VM finds and fixes the problem for you. Here's how:

- CheckOut/VM probes your VM system for down components
- CheckOut/VM *restarts* the down components it finds
- CheckOut/VM notifies critical support staff
- CheckOut/VM creates historical logs to summarize component failures

CheckOut/VM is easy to install. Full-screen menus create a friendly, interactive working environment for support staff.

"Check out" the newest availability manager from Duquesne Systems. Call **800 323-2600** (in Pennsylvania, call **412 323-2600**) or return the enclosed, postage-paid card to reduce your user phone call volume and staff workload today.

DUQUESNE SYSTEMS
Two Allegheny Center
Pittsburgh, PA 15212

***CIRCLE #1 on Reader Service Card* ▲**

Background

- VM Release: 04.42.0415
- Hardware: 3081 Mode = MP
- Memory:
  DPA: 27,209,728
  Free Storage: 4,713,096
  (Prime Free Storage: 184,320)
  Total Storage: 33,021,952
- Other Characteristics:
  - Paging areas are defined on solid state devices.

Data Characteristics

Certain properties of the data collected are particularly relevant to its analysis. In some cases, there can be problems with the data that cause its use to be questionable; in other cases, the way it was collected affects the way certain measurements are interpreted.

For example, the number of suspends reported by VMMAP is recorded. The VM monitor writes its data to a set of buffers maintained in free storage. If there are not enough buffers to hold the data, the monitor suspends collection and subsequent data is lost. A SUSPEND record in the monitor data indicates that such a loss of data has occurred. There are no suspends in the example, so we have some confidence in the quality of the data.

Data Characteristics

- number of suspends: 0

Load

One of the most important indicators for performance analysts and capacity planners is load: "how much the system is doing." In general expect service levels (that is response times) to degrade as load increases. Thus, service levels must be viewed in the context of load.

For example, the total number of logged on users, the number of active users and the virtual machine I/O rate are useful indicators of system load.

Load

- logged: 431.65 users
- active 128.31 users
- vio: 405.67 per second

Service Levels

Service levels, especially response times, are the most relevant indicators of system performance as perceived by the end user. For illustrative purposes, there are two commonly used VMMAP variables, Q1SEC and Q2SEC. (While Q1SEC and Q2SEC are considered unresponsive indicators of user service, their values in this example are significantly higher than expected. VMMAP includes several variables that provide better estimates of user response times, such as TRIVRESP and MINORESP. These estimates are more expensive to acquire than Q1SEC and Q2SEC, however).

The CP scheduler uses three queues to allocate the CPU resource. These queues are named Q1, Q2 and Q3. Short transactions typically complete their processing in Q1, while longer transactions are demoted to Q2 and eventually to Q3. With each demotion comes a lower dispatching priority.

Q1SEC indicates the amount of time that a transaction spent in Q1 before "dropping" — either to Q2, back to Q1 or out of the system. Q2SEC indicates the average time in Q2 (and Q3) before dropping.

FIGURE 2

Outline of VM Performance Worksheet

Analysis of VM System

1. Background
   - VM Release: ________________
   - Hardware: ________________
2. Data Characteristics
   - Number of suspends: __________
3. Load
4. Service Levels
5. Transaction Profiles
   - Overall based on resource queues
   - Overall based on user state samples
   - By Q1 drop (q1sec = _____)
   - By Q2 drop (q2sec = _____)
6. Resources
   - CPU
   - User I/O
   - Spooling
   - Paging and Swapping
   - Storage
   - Service Machines
7. Problem summary and Analysis
8. Recommendations
9. Questions

Service Levels

- q1sec: 0.59 seconds
- q2sec: 7.04 seconds

Transaction Profiles

A transaction's profile is a decomposition of its response time into component parts. By so doing, it is possible to focus on reducing the largest part to achieve the greatest benefit. For example, if you assume a homogeneous transaction mix (that is, the system is dominated by one type of work) and that the dominant workload waits for at most one resource at a time, then a profile based on measured resource queue lengths and utilizations can be derived. This profile does not provide response time information, only a percentage breakdown.

There are six resource areas that this decomposition identifies. Each field is expressed as a percentage of time spent using or waiting to use the resource:

- Run: using the CPU
- CPUQ: waiting for the CPU
- StgQ: waiting for memory (also called "eligible list wait")
- PgQ: waiting for page reads, directory reads and spool I/O to complete
- I/OQ: waiting for user I/O (also called "virtual I/O") to complete
- SwpQ: waiting for swap-in I/O to complete.

The resource queue decomposition for the example system is shown below. Note that other decompositions, such as by user, state samples and by Q1SEC and Q2SEC profile, are also effective.

Transaction Profiles

- Overall, based on resource queues (percentages)

Run	CPUQ	StgQ	PgQ	I/OQ	SwpQ
11	13	19	33	24	0

Resources

It is a maxim of performance analysis that delays are due to resource bottlenecks and resource bottlenecks typically result from high utilization of the resource. From the transaction profiles it is known that a VM waits for the following CP managed resources: CPU, memory (in the storage queue), paging (that includes spool and directory I/O), I/O and swapping. For each resource, utilization and how this can be reduced is assessed. Where possible, the impact that resource performance has on service levels is identified.

In addition, a transaction may be processed by a service virtual machine such as VTAM. So, although service virtual machines are not CP resources, it is appropriate to treat them in the same manner.

CPU

Total CPU utilization is indicated by the VMMAP variable SYSTCPU that can be broken down into SYSVCPU (CPU in "user" mode) and SYSCCPU (CPU in "system" mode). The ratio of SYSTCPU to SYSVCPU, called the T/V ratio in VMMAP, is a useful indicator of system overhead activity. Further breakdowns are

Resources

- CPU
  Total CPU: 197.94
  % Total CP CPU: 128.89 (T/V ratio: 2.86)
  % Due to page reads: 10.44
  % Due to spin locks: 4.5

possible such as determining the percentage of SYSCCPU spent handling page reads and the percentage of time lost due to spin locks.

User I/O

I/O delays are indicated by queuing for individual channels, control units and devices. The queuing typically results from a high utilization for each specific resource. In this section, channels, control units and devices with high utilizations or high queuing are reported. For devices, it is sometimes possible to identify the service level impact of high utilization as it is in the case of device IPLVOL in the example.

Resources
- User I/O
  - channel queuing is present on ch05, ch06, ch09
  - control unit queuing on controllers 500, 510
  - IPLVOL very busy at 54%; most activity on CP DIRECTORY
  - IPLVOL average time per I/O = 49ms (queuing time per I/O = 25ms)

Spooling

Spooling is an important activity for VM. Some applications make extensive use of spooling (that is, PROFS). Poor spooling performance has a direct effect on transaction response time. In addition, spool files consume important system resources even when their owners are not logged on.

For example, the spooling rate and average service time per spool I/O is recorded. Spool space occupancy is of interest because it can indicate a growth in the number of spool files in the system.

Resources
- Spooling
  - spool_I/O_rate: 24.6 per second
  - spool_I/O_time: .0251 seconds
  - spool space occupancy: 35%

Paging and Swapping

In a VM system, paging and swapping interact closely, so they are reported together in the worksheet.

In the example, overall measures of paging load and service such as the page read rate and average page service time receive the focus. Also, the decomposition of overall paging rate into the contribution made by each user ID is also effective for identifying paging problems.

Swapping has been disabled for this system; however, some swapping metrics are of interest. The logical swap percent, for example, indicates the percentage of logically swapped users who were subsequently physically swapped. Since swapping is disabled, these users will actually be paged out, so the percentage has significance for the paging subsystem. The interactive and non-interactive logical swap queue times indicate the average time that a page remains logically swapped. If these times become small, then it is an indication that logical swapping is losing its effectiveness in keeping user working sets in memory.

Resources
- Paging and Swapping
  - pageread_rate: 401.4 per second
  - pg_I/O_time:[3] .011 seconds
  - top 5 paging userids:

Userid	Pageread rate (per second)
USER01	8.9
USER10	8.9
USER03	6.3
USER22	5.4
USER04	5.4

  - no swap areas defined
  - logical swap percentage: 46%
  - interactive logical swap queue time: 3.9 seconds, non-interactive logical swap queue time: 0 seconds

Storage

Storage usage is complex to track and difficult to analyze. Fortunately, there are some key indicators that allow for ready identification of most storage related problems.

Correctly sizing free storage is an important tuning activity for VM/SP HPO (and VM/SP). Free storage and prime free storage are dedicated to the system for control block allocation. If free storage is too small, then the system will "extend" into the paging area to borrow more; if prime free is too small, the system will "miss" and borrow some free storage. It is important to examine the utilizations of these areas as well as their ability to satisfy system requests. For example, free storage and prime free storage utilizations as well as the free storage extend rate and prime free miss rate are recorded.

The steal rate is also recorded as this is an important indication of contention for the dynamic paging area. In the example, two steal rates are recorded: total

Resources
- Storage
  - Free storage utilization: 95.3
  - Extend rate: 4.3 per second
  - Extend depth: 11 pages average, 36 pages maximum
  - Prime free storage utilization: 50.0
  - Prime free misses: 0.0 per second
  - Steal rate: 0.05 per second, above 16MB steal rate: 0.00 per second

steal rate and previously 16MB steal rate. Both steal rates are recorded because it is important to identify where page stealing is concentrated, above or below the 16MB line.

Service Machines

Service machines are critical to many VM applications. They operate as disconnected virtual machines that await requests from users, process those requests and resume waiting. Unfortunately, the most useful information about these machines such as who used them, how long did each interaction require and what were the resource requirements for each interaction is not available. However, even without this information it is valuable to examine key performance indicators for important service machines.

For example, VTAM is one of the dominant service machines on VM systems. Its performance can be critical to the performance of all users of the system. We have recorded the VTAM delay profile — available directly from the VMMAP OUTSAMP report. It is also important to track VTAM's paging rate.

Resources

- Service Machines
  - VTAM:
    - Delay Profile:

Run	CPUQ	StgQ	PgQ	I/OQ
79	5	1	6	0

    - Page rate: 3.6 per second

Problem Summary and Analysis

This section presents an analysis of problems for the system. Rather than simply listing the problems, the key objective in this section is to communicate understanding about each problem. So, each problem is placed in context with explanatory text.

The example shows a portion of the complete analysis presented in the next section.

Problem Summary and Analysis

1. CPU Overhead is High:
   - CP CPU is very high indicating significant overhead activity. Some of this is due to paging, but the free storage extends and spin locks contribute.

Recommendations

Here the analyst records recommendations resulting from his earlier problem analysis. Where appropriate or possible these recommendations are specific; otherwise, a general recommendation is given.

The example shows a portion of the complete recommendation section, presented later in this article.

Recommendations

1. The existence of stealing (although not high) and free storage extends indicates that free storage requirements may be excessive. A reduction of free storage requirements may improve performance. There are several possibilities:
   a. Investigate the cause of free storage fragmentation.
   b. A large number of spool files puts a heavy demand on free storage. Determine the number of spool files in the system. Each spool file requires 128 bytes of free storage; 3,200 spool files use up 100 pages of free storage.
   c. Less than 50% of prime free storage is used. By reducing prime free by 25% you can return 11 pages to free storage. (Not much, but the average extend depth is 11).

Questions

While performing the analysis, many questions arise. Sometimes the questions pertain to system components over which the performance analyst has no control. Sometimes they are technical questions about the system that were not addressed in the VMMAP analysis and further data collection may be necessary.

Questions

1. DIRECTORY contention:
   a. What can be done about device IPLVOL? Are there ways to reduce contention on DIRECT?
      - Are there other ways to share?
      - What is all the access for?
      - Are the solid state devices volatile? (Can they be used to hold the DIRECTORY?)

In any event, the questions section of the worksheet can be one of the most useful. Perhaps this is because it both helps to focus future efforts and stimulates both the performance analyst and those with whom he communicates to think a little more about the system and its problems.

Details of Problem Summary and Analysis

Performance problem analysis lends itself to a systematic approach. This methodology can be described as the search of a tree of relationships between performance indicators. A small fragment of a search tree for VM/SP HPO is shown in Figure 3. This search tree is used in the analysis of transaction profiles. At the top level, it expresses the total number of users in system queues as the sum of the number of users queuing at or using CP resources, that is:

$$users\_in\_queue = \frac{SYSTCPU}{100} + CPUQ + STGQ + PAGEQ + I/OQ + SWAPQ$$

that is,

$$users\_in\_queue = 1.98 + 2.32 + 3.52 + 6.04 + 4.32 + 0.00 = 18.18$$

At the next level, PAGEQ is decomposed into the sum of users performing spool I/O, directory I/O and page reads, that is,

PAGEQ = spool_queue + directory_queue + page_read_queue

that is,

PAGEQ = .62 + 1.08 + 4.34 = 6.04

At the lowest level shown, spool_queue is decomposed into the product of spool rate and spool I/O time, that is:

spool_queue = spool_rate × spool_I/O_time

that is,

spool_queue = 24.6 × .0251 = .62

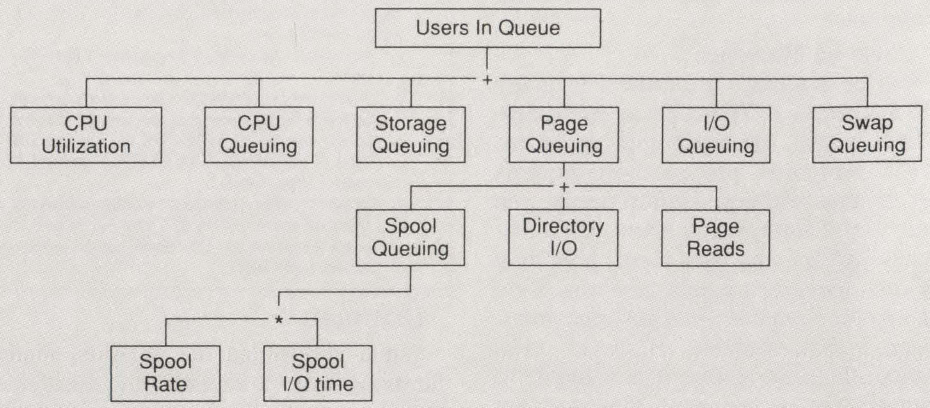
The examples above illustrate the ideal situation in which each node in the tree either corresponds to a measured VMMAP variable or can be decomposed into variables that do.

Once a problem is identified (that is, a user complains of poor performance, a user council observes a service level violation or perhaps the performance analyst notes that a threshold for a key metric has been exceeded), the following is done.

- The indicator associated with the reported problem is located. For example, this might be the user's response time or the performance analyst's metric of interest.
- Using the search tree as a guide, the next step is to break the problem indicator into its component parts. This step is called *decomposition*. Ideally, this decomposition is able to account for all of the indicator's value. For example, if the component values are added up, the result is the indicator value. In general, this is not the case and an approximation is the result.
- Each component is evaluated for exceptional characteristics, that is, check if 3380 device service time exceeds 25 ms. Leverage is often found in concentrating on the component that makes the greatest contribution to the total (that is, in the case of an additive relationship, this is just the largest component).
- The exceptional component(s) are then further decomposed. The process continues until the "leaves" of this decomposition tree are found. The leaves provide evidence for the existence of the problem. The path from the leaves to the problem may suggest an explanation for the cause of the problem.

FIGURE 3

Partial Decomposition Tree for VM/SP HPO



Unfortunately, there are some complications. Few indicators can actually be decomposed into mutually exclusive and collectively exhaustive components. In some cases, this is due to measurement problems, that is, the sum of CPU used by Q1 users and Q2 users never equals the total CPU consumed on the system. In other cases, as in "extend rate" for example, the algebraic relationships required for decomposition are not known. In these cases, factors that affect the indicator are identified. Where possible, note the directional effects that these factors have. For example, extend rate increases with increasing number of spool files, increasing prime free miss rate, increasing system demand for large blocks of free storage, increasing free storage fragmentation and increasing free storage utilization.

However, even without clear algebraic relationships, the search tree methodology can be applied. The analysis is less precise but can proceed in roughly the same way described above. Instead of simply looking for the largest subcomponent, look for supporting evidence for the existence of a particular factor. Frequently, these factors can be quantified, that is, "the prime free miss rate is 0.0," in which case there might be guidelines for interpreting their significance.

FIGURE 4

Example: Problem Summary and Analysis

Problem Summary and Analysis

a. A Storage Contention Problem Exists (PAGEQ + STGQ = 52%):
   - IPLVOL is busy; the DIRECTORY is heavily accessed. The substantial queuing on this device contributes an additional 24ms to each I/O. 18% of all users reported as waiting in the page queue are actually waiting at this device.
   - The storage queue times (eligible list) are high. This suggests that the system perceives a lack of pageable storage. All other indications support this. However, there is little page stealing that suggests storage is not constrained.
   - The reported physical swap percentage seems high. This number indicates the percentage of logically swapped users that are actually physically swapped. Since swapping has been disabled, the pages will actually be physically paged out. It is not clear what the system does but there could be significant overhead in processing swap outs that never happen. Low interactive queue time is also indicative of heavy memory contention.

b. CPU Overhead is High (CPUQ is large due to CP overhead):
   - CP CPU is high indicating significant overhead activity. Some of this is due to paging but the free storage extends and spin locks contribute.
   - Spin time of 45 ms/sec is too high and indicates a problem with CPU-AP contention. This contention is dominated by I/O lock (12ms) and Real Storage lock (29ms) activity. This is probably a symptom of heavy paging activity and demand for below-16MB storage.
   - Free storage is fragmented. On average, there are no large blocks available. This is causing frequent free storage extends to occur. The cost of a free storage extend is high and frequent extends should be avoided.

c. VTAM is paging at greater than one per second. This is a problem. VTAM is also experiencing significant CPU queuing delays. VTAM should receive the best service possible.

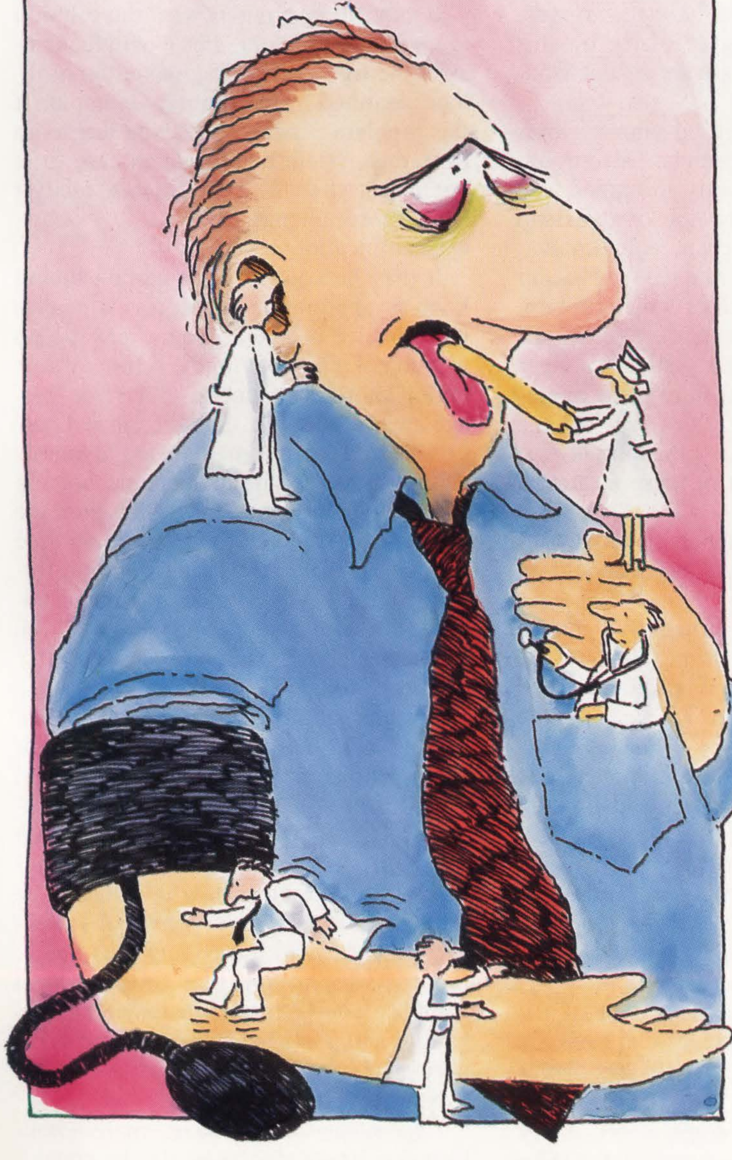
Example

The analysis presented in Figure 4 was developed with the search tree approach reviewed at the beginning of this section. The derivation of this analysis for each of the problems identified above is presented next. (For reasons of brevity, the analysis presented is not complete for this system. Only problems relating to storage, paging and CPU overhead are discussed.)

Storage contention analysis

Begin the search with PAGEQ, since it contributes 33 percent to the transaction

IS VM SYSTEMS MANAGEMENT HAZARDOUS TO YOUR HEALTH?



RESERVE YOUR PLACE TODAY

CALL **(800) 562-7100** OR **(703) 264-8413**

CIRCLE #5 on Reader Service Card ▲

Rx: THE VM SOFTWARE SEMINAR

Inflamed users. Swollen budgets. Paralyzed data. Fractured security. There's a lot that can happen in a data center. And a lot riding on your ability to diagnose and treat problems before they become critical.

Fortunately, help is available—and it's free. At the VM Software Seminar, you can learn first-hand how to improve your control over VM operations while simplifying data movement and SNA network administration across multiple environments.

SYSTEMS MANAGEMENT FOR TODAY AND TOMORROW.

This unique half-day event is packed with information on specific tools and techniques that enhance your ability to serve users while minimizing demands on data center personnel.

You'll see the comprehensive functionality that has made VMCENTER II the world's leading VM systems management package— your best guarantee of sustained superior performance and cost-effectiveness.

You'll also see a series of powerful products for moving bulk data quickly and easily throughout your SNA network. Including not only other VM sites, but other operating environments as well.

It's the perfect seminar for today's needs. And the perfect prescription for lasting success in an increasingly network-oriented world. So don't delay—reserve today. And join us for a healthy discussion of issues too important to miss.

SEMINAR AGENDA

8:30 a.m.	Registration and Coffee
9:00 a.m.	Seminar Begins
12:30 p.m.	Complimentary Lunch

SEMINAR DATES AND LOCATIONS

Atlanta, GA April 7
Austin, TX April 4
Boston, MA April 18
Chicago, IL April 20
Cincinnati, OH May 4
Dallas, TX April 5
Denver, CO April 28
Detroit, MI April 21
Edmonton, AB May 9
Los Angeles, CA April 25
Milwaukee, WI May 18
New Brunswick, NJ April 19
New York, NY April 11
Oakland, CA May 11
Orlando, FL April 6
Philadelphia, PA May 2
Phoenix, AZ April 27
Richmond, VA May 5
St. Louis, MO May 17
San Francisco, CA April 26
Toronto, ON May 3
Vancouver, BC May 10
Washington, DC April 14
White Plains, NY April 12

SYSTEMS CENTER
FORMERLY VM SOFTWARE, INC.

I'm sick of worrying about systems management.
Please enroll me at once in the VM Software Seminar

at ____________ on ____________
Name ____________
Title ____________
Company ____________
Address ____________
City ____________
State ____________ Zip ____________
Phone (______) ____________
Operating System ____________
CPU-Manufacturer/Model ____________

Systems Center, Inc., 1800 Alexander Bell Dr., Reston, VA 22091

Note: Attendance by a software vendor or its representative is prohibited without prior written approval from Systems Center, Inc.

A-MFJ-8903

profile. The reported page queue includes waits for directory reads, spooling I/O and real paging. It is not always simple to separate these components but in this case there is enough information to estimate.

- Directory reads constitute the dominant I/O to volume IPLVOL. Direct the search to this device and note that delays at this device comprise 18 percent of the overall reported page queue. This warrants further search, thus decompose the response time at the device into its components. The principle components and their contributions are: seek, transfer, latency and RPS (49 percent) and queuing (51 percent). The conclusion is that queuing on IPLVOL is contributing significantly to the paging component of transaction response time.
- Spooling I/O contributes in a smaller way to paging delays. An examination of spooling device response characteristics reveals no unusual contribution of any particular component. The conclusion is that spooling is not a significant contributor to poor performance.
- Real paging delays will increase if either page read rates increase or page read times increase. Page read times are less than 11ms since solid state devices are used. Thus, look at page rates. An absolute assessment of paging rates cannot be made since what constitutes a good value is not known. Therefore, assume that paging rates are too high and proceed further.

There are a couple of views of paging rates that are helpful. First, decompose them into rates by each user. In this case look for a dominant user to blame for the problem. There are no apparent culprits, so take an alternate view: look for factors that increase overall paging rate. The primary cause for increased paging rates is increased contention for storage and this is suggested by a few indicators.

- More than half of all logically swapped users are being physically swapped. The average time that a logically swapped user remains so is less than four seconds.
- Storage queue times are high. This suggests that the system perceives a storage shortage.
- The system is extending regularly. This means that page frames normally available for user use are being held by the system, thus increasing storage contention.

The conclusion is that storage contention has increased the paging rate.

CPU Wait Analysis

CPU queueing is significant in the transaction profile. Contributors to CPU wait include high non-overhead activity and high overhead activity. The high T/V ratio suggests that this system is experiencing high system overhead, so focus on causes for high system overhead. These include paging overhead, spin locks and free storage extends. It was previously recognized that storage contention has contributed to increased paging rates. It is reasonable, then, to assume that the increased storage contention is partially responsible for the increased CPU overhead. Spin locks seem high in this system. (The measured spin lock percentage is much higher than a recommended threshold value.) Another contributor to CPU overhead is free storage extends. Extends are occurring regularly on this system, so confirm extends as contributors to CPU overhead.

Continue decomposition with a closer look at free storage extends. Decompose free storage extends into an extend rate and an extend cost. The cost of an extend or how to change this cost are not known, so focus on the extend rate. Extends occur for many reasons some of which are: excessive spool file count, prime free misses, a heavy demand by the system for large blocks of free storage, infrequent migration of page and swap tables, free storage fragmentation and high utilization of free storage. We take each of these in turn in the analysis.

- Spool files: the number of spool files has not increased by a large amount on this day.
- Prime free misses: prime free misses are not occurring. Prime free storage may be too large but it has not changed in a while so this is not causing the problem.
- System demand: there are applications that can place heavy demand on free storage by requiring that the system allocate free storage in consecutive pages. None of these suspects is present on this system.
- Migration: migration seems to be occurring regularly on this system.
- Fragmentation: an examination of the free storage map indicates that there are almost no large blocks (blocks larger than 2K) available.
- Utilization: on the average, 214K of free storage is free. Free storage is not fully utilized.

The conclusion is that free storage fragmentation is a problem on this system. This fragmentation is causing frequent extends and these extends are increasing CPU overhead.

VTAM Analysis

VTAM is a critical subsystem. An examination of VTAM's transaction profile indicates it is paging at a rate greater than one per second. This indicates a problem and so is noted here.

Details of Recommendations

Making recommendations, unlike performing a problem analysis, is not an activity with a well defined methodology. Once a problem has been identified, finding a path to a solution is not always straightforward. There are several complicating factors.

- Identification

It may not be obvious what the solution is given the problem. For example, consider the free storage fragmentation problem identified in the running example. If this problem persists, it may be due to an operating system bug in which case, it is not clear what the performance analyst should recommend.

- Quantification

It may be known that a certain tuning knob is appropriate to address a certain performance problem. There may be no way to put a number on the setting of that tuning knob. For example, if the free storage area is too large, then the unused portion should be returned to the dynamic paging area for virtual machine use. In VM/SP HPO it is possible to measure the use of free storage, so determining free storage adjustments is more straightforward, but for VM/SP, the amount of unused free storage cannot be measured. An experimental approach must be taken.

- Effect

In some cases, it is not at all clear what the effect a tuning knob will have. For example, paging bias can be manipulated to affect the CPU priority that users receive. This will also effect the time that users spend in the eligible list. Unfortunately, it is not clear how this should be set, nor is it clear how sensitive system performance is to the paging bias value. In this case, paging bias interacts in a complex way with other parameters. It is hard to anticipate the effect of these complex interactions.

- Non-technical

Recommendations have many non-technical issues that must also be considered. These considerations complicate the process: financial, that is, "we can't afford any more type x devices;" security, that is, "department abc's data must be on its own volume;" policy, that is, "DMKSYS may be changed, at most, once a month;" political, that is, "only systems programmers can issue that command."

Even when such considerations exist, it is suggested that recommendations be communicated so that management is aware of the available opportunities.

This "methodology" is to focus on the problems identified in the problem analysis section. Then the following approaches are used.

■ The standard fix, or literature search

Certain problems have been identified as being common and important for VM systems. In many cases, reports have been written describing these problems and their solutions. This is the first place to look.

■ A model based approach

If a problem has been identified with the search tree methodology and using algebraic decomposition, then it is possible to use the algebraic decomposition to focus on specific recommendations. For example, consider a problem relating to response time that is further decomposed to reveal that the underlying problem is excessive CPU queuing (see Figure 3). A high level recommendation would be to "reduce CPU queuing."

Also, see the notes attached to the example in this section.

■ An experimental approach

Whenever it is hard to quantify recommendations or even gauge their effect, an experimental approach is necessary. For example, on a VM/SP system it is impossible to measure the degree of free storage utilization, so free storage adjustment must be done experimentally. Suppose free storage extending is occurring: increase free storage by 10 percent and measure free storage extend rate, if non-zero go to one.

At the conclusion of this investigation, free storage will be just large enough to avoid extends which is just where you wanted it to be.

Unfortunately, an experimental approach may not always be possible or feasible. The changes that must be made might need to be performed by someone in a department different from the performance analyst. These changes might have to be scheduled as infrequent tasks, rendering it difficult to quickly converge on a reasonable setting. In these situations, it is still valuable to consider an experimental approach. However, more time will be needed.

■ Question unusual situations

It is valuable to question anything that seems unusual. This is good advice for problem detection as well. Performance analysts develop good intuition for their systems and any unusual event warrants closer inspection.

Often, decisions are made about system management that become outdated. Systems are complicated and people are so busy that there may not be time to review old decisions. Suggesting a re-evaluation of relevant policy decisions is sometimes a reasonable approach. At the very least, it inspires people in the organization to rethink why certain decisions were made.

Example

The derivation of the recommendations shown in Figure 5 is presented.

Recommendation One

A queuing problem on device CBVIPL is identified. This problem is due to the heavy utilization of this device, specifically the DIRECTORY area. The recommendation to reduce I/O rates or service times is based on an algebraic relationship describing device utilization:

utilization = I/O_rate × I/O_service_time

So, a reduction in either I/O rate or I/O service time will reduce utilization and the queuing problem. The recommendation is not specific as to how to achieve this reduction. There are several ways to do this. There are three approaches to reducing I/O rate. The first two attempt to move I/O activity; the first in time, the second in space. The third approach attempts to eliminate some of the I/O activity.

- Off-load some activity by moving it to another shift. In order to do this, it will be necessary to identify major users of the DIRECTORY and investigate some shift changes.
- Off-load some activity by moving some minidisks/system areas to another (less busy) device. (The directory activity cannot be divided, so that has to stay in one place.)
- Reduce the need to go to the directory so often. In order to do this, find out which applications access the directory and why. Perhaps this will lead to something that can be done to reduce the rate.

Both approaches for reducing I/O service times also rely on an algebraic model for determining the recommendation.

- One approach is through tuning but IPLVOL's service time of 24ms is not unreasonable for a 3380, so little benefit is expected.
- Another way of reducing I/O service time is to put the data on a faster device. This system has several solid state devices that are used for paging. These devices have an average service time of less than 10ms. If it were possible to put the directory on one of these devices, then significant benefit would be realized.

Recommendation Two

Some fixes were introduced to VM in 1986 that attempt to alleviate some of the cost of storage contention below the 16MB address line. A feature of these fixes is the use of storage above the 16MB line as an alternative to paging out the contents of below 16MB line frames when those frames are needed. The VMMAP variable LOIII is a measure of the volume of pages that are being moved from below 16MB to above 16MB. For this system, this value was 0, so the recommendation was made to ensure that the fixes were installed.

FIGURE 5

Example: Recommendations

Recommendations

a. Try to either reduce I/O activity to or reduce I/O service time of IPLVOL. Consider moving everything, except DIRECTORY, off that volume. Identify cause of all I/O activity to the directory. See the questions below.

b. Ensure that APARs VM24485 (PUT 8604), VM25601 (PUT 8605) are installed. These fixes allow the system to make better use of storage above the 16MB line when storage below the 16MB line is constrained.

c. The existence of stealing (although not high) and free storage extends indicates that free storage requirements may be excessive. A reduction of free storage requirements may improve performance. There are several possibilities:

1) Investigate the cause of free storage fragmentation.

2) A large number of spool files puts a heavy demand on free storage. Determine the number of spool files in the system. Each spool file requires 128 bytes of free storage; 3,200 spool files use up 100 pages of free storage.

3) Less than 50% of prime free storage is used. By reducing prime free by 25% you can return 11 pages to free storage. (Not much, but the average extend depth is 11).

d. Consider enabling the swapping system. The I/O configuration seems capable of supporting swapping. 3380s are excellent swapping devices. The solid state devices could be used for paging and to relieve contention on critical volumes (e.g. those mentioned above).

e. It is possible that the high storage queue is artificial. With fast paging devices you can support more users, but the system may not appreciate this and hold users in the eligible list. Consider increasing APAGES or decreasing the PAGING BIAS to regulate the system's use of the storage queue.

f. VTAM is paging too much and needs to be tuned. It has 760 pages reserved but maybe that isn't enough.

Recommendation Three

Free storage extends are a serious problem. In the problem analysis section several possible causes for the extensions were identified and the principle cause received the focus: free storage fragmentation.

- The natural recommendation is to "reduce free storage fragmentation." Unfortunately, it is not clear how you do that. If users do not log off frequently enough, then free storage can become fragmented. However, it seems that this is not the case here.
- Another approach is to attempt to reduce free storage consumption. In the problem analysis this was discounted as being the principle cause of the extend problem but a reduction in storage requirement would probably also benefit the fragmentation problem. Spool files are a significant consumer of free storage so a reasonable recommendation is that the number of spool files be examined.
- Along the same lines, note that prime free storage is less than 50 percent utilized. Unused prime free storage is wasted so it should be redefined as free storage. It is interesting to note that the amount of prime free storage that is unused is the same amount of free storage that is extended (just a coincidence). This suggests that returning the prime free storage to free storage might help.

Recommendation Four

The decision to disable the swapping system is an unusual one for VM/SP HPO 4.42. This is pointed out and whether the decision is still reasonable is questioned.

Recommendation Five

Eligible list times are quite high in this system. Adjusting the eligible list is one of those mysterious functions in VM for which there are no algebraic relationships with which you can feel comfortable. Instead, it is known that APAGES is recommended as a tuning aid for manipulating the eligible list. It is also known that the PAGING BIAS plays a role in the determination of time spent in the eligible list.

The recommendation given here is speculative. It assumes that the eligible list wait is artificial and attempts to force the system to let more users into memory. In fact, this recommendation might not work at all. Some other knobs might prove more successful.

Recommendation Six

VTAM is paging at more than one per second so a paging problem is diagnosed. One of the recommendations in the report is reserving a certain number of pages for the VTAM service machine. It is evident that this has been done already but does not seem to be holding down the page rate.

Conclusions and Future Work

The worksheet methodology has been applied to more than 30 VM systems of all sizes, ranging from a 9373 Model 20 running VM/SP to a 3090 Model 200 running VM/SP HPO 4.42. In all cases, the worksheet has proven a useful tool for the summary and communication of VM per-

CIRCLE #59 on Reader Service Card ▲

formance information. The worksheet based methodology achieves many valuable goals for the performance analyst.

Time saving: the VMMAP report must still be looked at, but the worksheet provides a focus for the extraction of the valuable information.

Communication: the worksheet is relatively concise. Other individuals in an organization can quickly locate the portions of the worksheet that are relevant to them.

Organization: the worksheet is organized in a way that is complementary to the systematic approach taken by many performance analysts.

One of the motivations for developing the worksheet was to save some time for the busy performance analyst. The worksheet provides a methodology so it certainly accomplishes the goal. However, an automated aid will save even more time. It is clear that certain portions of the worksheet lend themselves more readily to automation than do others. Specifically, the first sections that deal with the extraction and organization of key indicators for key resources are readily automatable. This is where the automation process was begun.

The last sections, specifically "problem analysis," "recommendations" and "questions" are much more difficult to automate. The search graph methodology discussed for problem analysis suggests an approach for automation of this section. However, this is not so easy considering problems like exception identification. For reasons discussed earlier, recommendation generation is even harder. For certain kinds of problems, especially those involving the I/O subsystem, automating the generation of recommendations is more manageable.

ABOUT THE AUTHORS

Joseph Hellerstein is a research staff member at IBM's T. J. Watson Research Center in Yorktown Heights, NY. Prior to completing his Ph.D. at UCLA, he worked at Honeywell Inc. for several years designing operating systems, working in communications and networking software. Since joining IBM in 1984, he has worked on expert systems for performance management and VM performance.

Dr. Robert Berry is a research staff member at IBM's T. J. Watson Research Center in Yorktown Heights, NY. He earned his Ph.D. degree from The University of Texas at Austin in 1983. Prior to joining IBM he was employed by Boole & Babbage, Inc. designing capacity planning and performance management software. Since joining IBM in 1987, he has worked on performance tuning and performance problem diagnosis techniques for VM.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
230 (Interesting/Helpful), 231 (Too Technical), 232 (Too Basic)

2.) Would you like more articles on the same subject?
233 (Yes), 234 (No)

Capitalize On Greater Performance

You don't have to be a member of Congress to meet distinguished colleagues in Washington, D.C. At the third annual Candle Performance Conference this spring, you'll be part of a history-making agenda. It's the premier forum for the performance and tuning experts who run today's high-performance online systems. The conference offers more than 60 sessions under one roof, all for the price of a single conventional class.

Watch Performance Blossom

Only the 1989 Candle Performance Conference offers this much *immediately applicable* information on system performance...insights that can quickly boost your data center's productivity and service levels. In addition to the environment-specific sessions, you'll also benefit from the scores of in-depth seminars on networking, automated operations, and performance management trends. Your session instructors are all industry veterans with years of hands-on experience. And because the focus throughout is on learning, not note taking, detailed conference transcripts are distributed free to all participants.

Don't Filibuster — Enroll Now for Special Discounts!

The conference fee of $1,500 is a real bargain because it includes admission, documentation, and all hosted meals. And you'll save $200 if you register before March 30, so don't let your savings get hung up in committee. Enrollment is limited. For registration details, call the Candle Education hotline today at (213) 442-4075.

Candle®
Candle Education
1999 Bundy Drive
Los Angeles, CA 90025

April 30 — May 5, 1989

Hyatt Regency Hotel
Bethesda, Maryland

CIRCLE #116 on Reader Service Card ▲

How Dyslexia May Affect Data Processing Professionals

By Ted C. Keller

One of the most meaningful moments of my life occurred when I was about 25. While watching a news commentary, I learned that Nelson Rockefeller, governor of New York and later vice president of the United States, was bothered by an affliction called *dyslexia*. This affliction, the commentary said, caused letters and words to appear out of order. Instead of seeing things as they were printed, the dyslexic individual would see things in a scrambled pattern. The commentary indicated that dyslexia could be overcome and that Governor Rockefeller was a good example of someone who had done this. This was the first time that I realized there was such a thing as dyslexia and I recognized almost immediately that this was a problem I had been fighting most of my life.

Dyslexia or more properly, *functional dyslexia* or *specific language disorder,* is a neurological dysfunction affecting visual, auditory and spatial perception. The most commonly publicized aspects of dyslexia have been the perceptual problems associated with reading and language disorders but dyslexia is much more than this. Dyslexia involves a wide range of perceptual difficulties in which information from many of the body's sensors arrives in the brain in a confused state. Individuals suffering from dyslexia may experience confusions of time and space, visual perception and auditory understanding.

While dyslexic individuals face many challenges, they also tend to have above average intelligence and good mathematical and reasoning skills. Since these are the very skills most commonly shared by data processing professionals, it is quite likely that a fair percentage of those reading this article may suffer from this dysfunction as I do. It is my hope in writing this paper that others who share this problem will benefit from my experiences and come to a better appreciation of themselves.

My Early Experience With Dyslexia

When I was younger, I was somewhat uncoordinated (a sign of dyslexia) and had a strong left-right disorientation (a sure sign of dyslexia). I can remember not being able to tell my left hand from my right hand until I was about eight or nine years old. I can recall my stepfather suggesting that I just remember which hand I used for writing. When I tried to do that, my mind would go blank; but when I would later pick up a pencil without thinking about it, I would naturally use my right hand. When I was about eight years old, I injured my right wrist. The injury left a rather large, conspicuous scar. From that point on, I always could tell which was my right hand — it was the one with the scar on it. Eventually this relationship became firmly engraved in my mind and this specific disorientation ceased to be a problem.

Throughout my life there have been other similar confusions that have challenged my creativity and patience. It used to be difficult to remember which faucet controlled the hot water and which the cold, especially if they were not marked. It was difficult to remember that most threaded things were turned clockwise for "in" and counterclockwise for "out." Other relatively simple relationships eluded me until I worked with them enough that firm patterns could develop in my mind. For the most part, these aspects of daily life no longer trouble me unless I am very tired or under a lot of stress.

I can also remember that as a child I had a hard time differentiating between the letters *b* and *d*. I also had trouble remembering which way letters like *S* and *N* faced. I do not remember much about this phase of my past except that it was difficult to understand why I could not seem to learn such things while others could do so easily. Suggestions like, remember that the bump is on the left side for *d*, were no help since it would be quite a while before left and right became firmly cemented in my mind. I suppose that my learning came mostly after much drilling with various self-created memory aids. Somehow I eventually forced myself to associate the appropriate shapes with the correct symbols.

Throughout grade school and into junior high, "mathematics" was very difficult for me. When I was going to school, *mathematics* was almost entirely *arithmetic* through junior high. It seems that the major difference between fifth and eighth grade math was the size of the numbers we were expected to multiply or divide. While I had no difficulty remembering the various multiplication or division tables, a series of numbers would not always be perceived in the same order in which they were written. Even though I could do a reasonably good job of determining the solution, it had to be a slow process in order to allow proper perception of the various digits involved. And, even if I could correctly calculate the answer, there was a reasonable chance that I would record it incorrectly. I suppose I must have appeared rather dull to those responsible for teaching me arithmetic. I can remember being told that I was passed in some of my early years not because of what I could do but because my teachers seemed to think I knew the material well enough to advance.

In high school, my world changed. In ninth grade I took a class in elementary algebra; this was the beginning of the rest of my life. I enjoyed playing with symbols and algebra fascinated me. I started reading ahead in the text and working problems on my own. Within three months I had mastered not only the material in the freshman text but most of that in the junior algebra text as well. Before mid-year I was working on college algebra and calculus. It seemed that I was constantly exploring my new world working math problems of some kind. Like many other dyslexics, I had been blessed with strong relational and analytical skills. For the first time in my life I discovered something at which I could excel and this gave me a chance to grow out of my shell.

 CIRCLE #3 on Reader Service Card ▲

Courses like English and history continued to plague me throughout high school. I was always a slow (if not very slow) reader and these courses always required much reading. Unless I read slowly, the words might look like other words or appear in a scrambled order. I can remember not keeping up with reading assignments, partly because of my slowness and partly because of my unwillingness to spend much time doing that which did not seem productive.

Dyslexia and the Business World

When I first started working as a programmer, I quickly found my niche writing Assembler programs. The small number of characters and large reliance on symbolic relationships fit my patterns of strength quite well. Working in languages such as COBOL, in which a large amount of verbage was involved, not only did not appeal to me but could also have been disastrous. The likelihood of my mistaking or misinterpreting longer labels and words would have made it difficult for me to be as efficient as I could be in a more basic language. Luckily, I was able to work in areas suited to my strengths, finding both satisfaction and success. Even though I was not yet aware that there was such a thing as dyslexia, I instinctively concentrated on the type of work for which I seemed best suited.

My slowness in reading would haunt me most of my career. Not realizing the nature of my problem until several years after I began working as a programmer, I just considered myself a slow reader. While others seemed to be able to read a page in a minute or two, it would take me significantly longer. Concentrating on a word or two at a time, I could read words correctly and in the order in which they were written. If I looked at something quickly, what I saw was often not what was actually there. Because of the time needed to read any material, I had to be selective of which manuals I would invest time reading. This limitation forced me to learn to specialize and become proficient in selected areas. It also forced me to leave some aspects of my profession on the periphery of my awareness.

Knowing that I would never have the time to read general information about many things, I chose to concentrate on a few and become the best I could be in my chosen specialties. Over time, as my horizons expanded, the depth of my knowledge in many areas became a tremendous asset. But while being the "expert" in some areas, I would remain almost illiterate in others for long periods of time.

To overcome my inability to read large amounts of material quickly, I learned to shy away from tools or products that would require much reading. I would find something particularly useful to me and, partly as a result of the slowness with which I read, totally absorb most of its finer points and intricacies. When I did not have the time to do all the reading necessary to become familiar with some product or feature I needed to use, I would avail myself of the expertise of others. In time, I was able to develop a knack for asking questions and could usually draw a considerable amount of information from the right individual in a relatively short amount of time. (Of course, the trick was in locating the person with the right answers.) Balancing the experiences of others with selected reading, I could become proficient in most things quite rapidly.

Part of every programmer's job is the verification of test results. Whenever possible, I would let the computer do much of the verification work for me. I could allow technology to help me do those things I found most difficult.

Over time my reading skills seem to have improved. It seems that I can read somewhat more rapidly than when I was younger. The confusion of letters and words no longer seems as intense as it was in the past. While I probably will never be able to read rapidly, I have grown to a point where I am not as limited as I used to be. I still have perceptual problems if I look at something quickly but I can overcome this by taking my time.

One of my greatest challenges has been reading in front of other people. I have always been somewhat embarrassed by my slow reading speed. Whenever someone would bring me a page or two to read while they waited, I would panic mentally. Being somewhat impatient (especially with myself), my vanity would not allow me to admit to others how long it would take me to read what was before me. As a result, I would try to scan the material rapidly, gleaning whatever truth I could from the more significant keywords and a few key sentences. It took me years to realize that I would never be a speed reader and that it was all right to read things slowly. I have found that most people do not need a response to material immediately and usually do not mind if I read the material by myself and get back to them later.

Part of every programmer's job is the verification of test results. Typically, it is necessary to produce lists of input and output data and then manually inspect the output comparing it to expected results. This often involves comparing columns of numbers, names or text to similar information derived from other sources. With a propensity to reverse letters as I looked from one source to another, the task of verifying test data was always cumbersome.

As with other problems, I was eventually able to develop techniques that would allow me to do this reasonably well (but never very rapidly). One trick I found was to read the numbers or letters from each source aloud. (The involvement of multiple senses is encouraged as a way of overcoming many of the effects of dyslexia.) I would often see something one way and find myself reciting it in a different order. Somehow I was able to sense the difference between what I saw and what I said and compensate for my inadequacies. Over time these skills have improved and I seldom have much difficulty reviewing data as long as I limit myself to eight to ten characters or digits at a time.

Whenever possible, I would let the computer do much of the verification work for me. If I could find the same data in two different sources, it was usually a relatively easy task to write a quick program to match and compare data and display the results, providing a more thorough means of verifying data. I could allow technology to help me do those things I found most difficult.

Dyslexia also affects auditory perception. Though my hearing is adequate (except for very high frequencies), word sounds often become confused. While I

Without The VITAL SIGNS™ VM Performance Monitor, You Can Relate CICS Performance To VM Resource Usage For Improved Service and Planning...

But It *Isn't* Easy!

Now you can further improve service to end users and better plan growth by combining VM and CICS performance data. Quickly and accurately. When VITAL SIGNS™ is used in conjunction with Landmark Systems' The Monitor for CICS.™

VITAL SIGNS is the *first and only* VM performance monitor to include an interface to a CICS performance monitor. Through the interface, CICS performance data from The Monitor for CICS can be combined, by date and time, with VM data in the VITAL SIGNS performance database. Using the VITAL SIGNS Report Writer and Plot Generator, CICS performance information (e.g., response time) can be accurately associated with coinciding VM performance data (e.g., total system CPU utilization) to provide true correlations for identifying performance problems and trends. No more manual comparisons. No more guessing. And that's just the beginning!

VITAL SIGNS is also the *first and only* VM performance monitor to include DASD modeling. It allows you to perform "WHAT IF" analyses for such things as verifying the need for DASD upgrades, determining the effect of different file block sizes, and much more.

Of course, VITAL SIGNS also has both real-time and historical reporting facilities fully integrated into a single comprehensive product sold on a site license basis. You can have all major performance measures in both tabular and color graphics formats. Comprehensive DASD seek analysis facilities, such as a Volume Seek Map, are included. Exception reporting facilities provide early warnings of approaching problems. And an easy to use data extraction facility lets you pull specific data from VITAL SIGNS for further analysis by popular statistical or spreadsheet products on PCs or mainframes.

To find out first hand why VITAL SIGNS is more than just a VM performance monitor, call us today at ***800-826-0313*** (***612-542-1072*** in Canada and Minnesota).

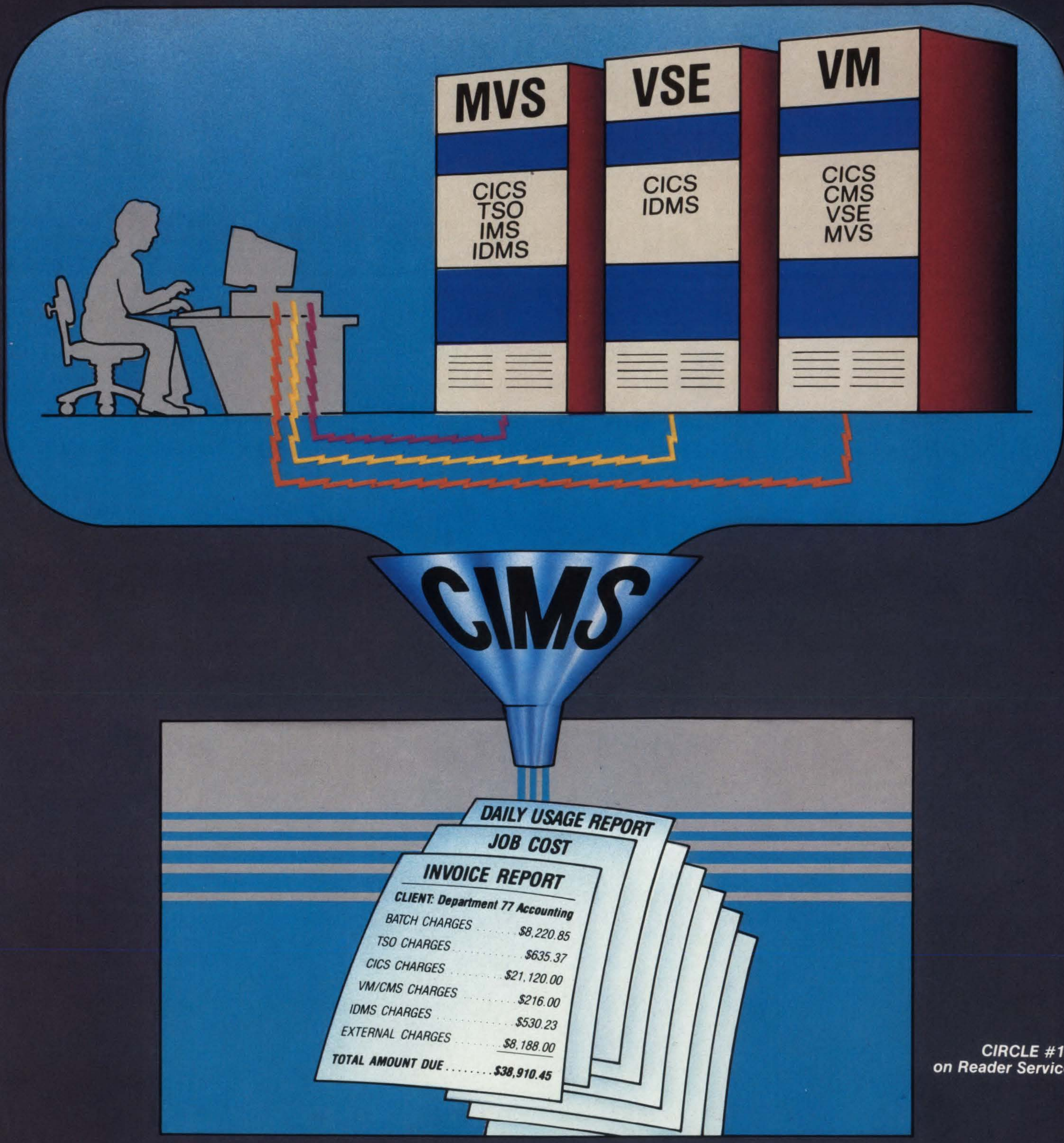
BlueLine™
SOFTWARE INC.

New Advances in Performance, Productivity and Planning.

CIRCLE #48 on Reader Service Card ▲

1500 South Lilac Drive, Suite 340
Minneapolis, MN 55416
800-826-0313
612-542-1072 (In Canada and Minnesota)

CIMS — *The Integrated Chargeback System*



CIRCLE #138
on Reader Service Card

Job Accounting • Chargeback • Cost Analysis
Resource Utilization Reporting

Call for additional information or a demo.

BMS Computer Inc. • ***Quality Products Since 1974***
375 North Wiget Lane, Suite 210 • Walnut Creek, CA 94598 • Tel. (415) 938-2620 • FAX (415) 946-0289 • Telex 469516

can hear the sounds distinctly, they do not always arrive in my mind in the correct order. Sometimes letter sounds become confused. When I hear a new word (especially if it is a long word), it may take me several attempts to pronounce it correctly. Though recognizing the sounds, I have difficulty retaining their correct order.

All my life I have had trouble hearing, not because of mechanical problems with my ears but because of a dysfunction that occasionally confuses order. And until I learned a few tricks, this problem hampered my communication with others. The first and most important technique I learned was active listening. This involves listening with one's mind as well as one's ears. It requires using attentiveness skills and resisting internal distractions. In order for me to comprehend what is being said, I must not only listen carefully but I must take advantage of visual and contextual clues. Then, when word sounds are occasionally transmitted incorrectly to my mind, I can sort them out and determine what was probably said. When listening in context, it becomes easier to follow what is being said.

To use these listening skills, I have learned to position myself so that I watch the person speaking. When involved in a conversation with more than one person, I try to allow myself to see as many of those speaking as possible. If I really want to pick up what is being said, I need to be more active and follow the conversation with my mind and body as well as my ears. Of course, I do not always have the energy to pursue this as diligently as I should, causing me to miss out on parts of some good conversations.

One last difficulty that I believe to be associated with dyslexia is my inability to recognize faces. While I have not been able to locate reference material relating this problem with dyslexia, it has just recently occurred to me that shapes and patterns in faces are little different than symbols used in language. It seems reasonable to me that if my neurological dysfunction could affect the way I perceive symbols, it could also impact my ability to recognize faces.

It has always been embarrassing for me to meet someone and shortly forget their face when meeting again. While I have no difficulty seeing features clearly, my mind has a difficult time compiling the series of images that constitute a face into a meaningful likeness of an individual. I have learned to study others' faces more carefully and diligently. I have also learned to take advantage of other non-verbal clues (such as voice or mannerisms) but this is not always effective. And, I have learned to develop contextual clues to help associate faces with the environment in which I am most likely to encounter them. This can be effective except when I meet someone I should know in a situation in which I did not expect to find them. I might be hopelessly lost trying to remember whether I recognized them and in what connection.

Dyslexia's Other Effects

Along with the physical limitations of dyslexia, individuals with dyslexia may develop behavior patterns to compensate for the inconsistencies they experience. In the past, when dyslexia was not well known or recognized, dyslexics would often be labeled as slow learners, lazy or unintelligent. Imagine what kind of self-image a child will develop when he or she cannot recognize basic letters, remember what direction the letter *s* faces, tell which direction is left or right and has trouble with other basic relationships. Imagine further how a lack of coordination can contribute to an already low self-image. (Some of the perceptual problems associated with dyslexia can lead to coordination problems with basic tasks — such as riding a bicycle or catching a ball.) The confusion of left and right can lead to difficulties with balance, steering, catching and so on. Before dyslexia was commonly recognized, the life of a dyslexic child could be filled with frustration and discouragement.

Everyone is an individual and will respond somewhat differently to similar situations. In some cases, especially when dyslexia is not diagnosed until later in life, the child may feel alone and inadequate. Children tend to pick up quickly on peculiarities in one another and can be quite cruel to others who are at all different. The dyslexic child's reaction may be to become introverted, turning inward to escape failure and ridicule. In many ways the dyslexic may have the same needs for achievement and friendship as other children but perceptual problems may interfere with chances of success. Depending on the child's needs and the seriousness of his or her disability, significant levels of anxiety can develop. Various kinds of avoidance behavior may develop as the young person tries to cope with a world that is hard to understand.

Most dyslexics can overcome many of their perceptual limitations with training and perseverance. Recognizing dyslexia's symptoms and understanding some of its mechanical implications can go a long way in overcoming frustration. Until I came to an understanding of my limitations, I used to be so terribly frustrated that I had difficulty doing some things that most others could accomplish easily. Having recently come to a better understanding of many of the features of dyslexia and how they have caused me so much trouble, I have gained a new appreciation of myself, my physical limitations and the learned patterns I have acquired to deal with my difficulties. For example, I can now see that I have always been shy and somewhat reluctant to meet new people because I was afraid that I would be embarrassed by not recognizing them later or be unable to understand them due to my limited auditory perception. I can also now understand my tendency to work alone, avoiding potentially embarrassing situations that might be caused by my limitations. While I am already learning new ways to compensate for and overcome the physical deficits of dyslexia, it may take time to adjust my life to take advantage of my newly attained insight.

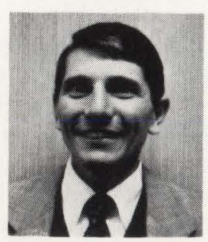
Conclusion

Dyslexia is a problem that affects many people, both young and old. It is not limited to any race or culture, nor is it directed more at any one socio-economic group than another. With more public awareness over the past several years, dyslexia is commonly detected early in childhood when remedial treatment is most effective. More than 20 million people suffer from some of the limitations of dyslexia and most can lead normal lives. Dyslexia is a problem that can be overcome but only when there is an awareness of its symptoms.

I suspect that many who read this article will recognize these symptoms either in themselves or someone close to them. Some may realize for the first time that difficulties they have been experiencing are merely signs of dyslexia. For many, the problems will be relatively minor as they have been in my case. In others, the problems may be more severe. In most cases, though, dyslexics can learn to overcome their limitations once they recognize and understand them.

More than anything, young dyslexics need love and encouragement. They need to find their strengths early in life and build on them. They need successes and chances to leverage their strengths. Those who have grown up with dyslexia without being aware of the nature of their dysfunction may need understanding, both of themselves and by others. They may need to learn to appreciate themselves so that they may look beyond their limitations and exploit their strengths. Dyslexia is a limitation but it can be seen as an inconvenience to be overcome or an obstacle in life. Hopefully, this article will play a small part in letting others become all they are meant to be.

ABOUT THE AUTHOR



Ted C. Keller is the manager of a group responsible for CICS systems support, performance management and capacity planning at Yellow Freight System, Inc., 10990 Roe Ave., Overland Park, KA 66211, (913) 345-3274. He has worked in various data processing jobs for more than 21 years.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
235 (Interesting/Helpful), 236 (Too Technical), 237 (Too Basic)

2.) Would you like more articles on the same subject?
238 (Yes), 239 (No)

NOREX Accelerates VSAM Processing;

Trims Processing Time By Half

Objective: Reduce the processing time required to run a mission-critical leasing system.

Solution: Accelerate VSAM with a high performance, indexed access accelerator.

By John Kador

When Honda Bhyat was appointed assistant vice president at NOREX Leasing, Inc. (Toronto, Ontario), the largest leasing company in Canada was facing a processing problem: the time required to process its mission-critical leasing application threatened to exceed the available batch window.

Sustained healthy growth in its leasing business created an increased volume of work and enormous pressures on NOREX's MIS department to speed up the processing of its leasing system. However while working faster was certainly a goal, getting bigger was not. An upgrade of the CPU, an IBM 4381 Model Group 14 running under MVS/XA, was quickly rejected as not financially acceptable.

NOREX found the solution to its problem by replacing VSAM with an alternate way to access files: IAM from Innovation Data Processing (Little Falls, NJ), the same company that provided FDR/ABR, the DASD management system used by NOREX. Under VSAM, according to Bhyat, run times for the leasing application averaged six to eight hours a night, a figure that was creeping steadily upwards. After installing IAM, the average run time immediately fell to three to four hours, a 50 percent reduction. Here was just the kind of relief NOREX needed for its batch window problem but there was another side to the system that just could not be ignored.

NOREX's leasing system, one of the most advanced of its kind, provides extensive on-line services for its users under CICS. The CICS portion of the system, in fact, relies almost exclusively on the same files that had been converted to IAM for the batch cycle. Moreover, the batch implementation had been undertaken with some apprehension as to what the impact of IAM would be on CICS.

What NOREX found was an unanticipated but very pleasant surprise. Simultaneously, terminal response time was slashed in half. Systems Management Facility (SMF) statistics revealed there were only half as many I/Os processed as compared to before IAM was installed.

Bhyat is quick to note that NOREX's processing constraint was not a function of VSAM per se. The way NOREX used COBOL contributed to the situation. In 1983, NOREX had converted from DOS/VSE to MVS. The leasing application developed under DOS in COBOL made use of VSAM in a way that was resource-expensive under MVS. Specifically, the leasing system was designed to make use of the ''Access Is Dynamic'' COBOL statement, a statement that allows programmers freedom to switch between processing modes. Under DOS, use of the statement incurred no significant penalty. Not so under MVS with its scrupulous integrity rules.

Nor was it practical to rewrite the application to eliminate the dependence on the statements. The potential rewrite was calculated to require six to eight man-months of coding, not to mention the time required for the exhaustive testing that would have been added to the cost. It was something no one wanted to undertake. "After looking at the problem in depth, we saw that even if we did the rewriting, the gain would be minimal," Bhyat says.

Optimizing VSAM

Leaving no stone unturned, Bhyat first thought that fine-tuning VSAM was the place to start. In fact, by optimizing some JCL, NOREX was able to extract some significant performance gains from VSAM. However with increases in workload, even with that increase in performance, running the leasing application threatened to overwhelm the available batch window.

Since VSAM fine-tuning helped but did not offer a permanent remedy, "There was simply no alternative but to actually speed up the jobs," Bhyat recalls. VSAM was a major bottleneck. NOREX looked for a higher-performing alternative. To be considered, the alternative had to offer at least all the functionality of VSAM, be faster, incur no greater overhead and be transparent in use to programmers and end users alike. "That's where IAM came into play," he adds.

After testing IAM, it was clear that NOREX had the solution to its problem in hand. On the batch side, the use of IAM resulted in a 50 percent reduction in run time over that achieved by fine-tuning VSAM. While for most jobs savings was 50 percent, savings on some heavy processing jobs were much higher. "The processing time required to run one job dropped from 14 hours to two hours," Bhyat notes. "Considering that our processing volume had increased, we'd require six hours of run time to process the leasing application. IAM has now knocked that down to three hours. Our month-end runs used to take two days. Now that takes one day. More work gets done in less time. We are able to sustain our promised service levels even with increasing volumes," he adds.

As if these results were not enough, NOREX actually concluded it could quickly justify IAM on the basis of the DASD space saved on test files alone. NOREX found it recovered anywhere from 35 to 50 percent of the DASD units its test files were consuming. "We were able to justify IAM on the DASD-savings basis alone without even putting it into production," Bhyat claims.

The data compression savings translated into substantial savings in magnetic media, as well. Under VSAM, NOREX used 28 reels of magnetic tapes nightly to do selective database backups. After it switched to IAM, it was immediately able to perform backups using only 20 tapes a night. Besides a savings of more than 200 reels of tape a month, there were corresponding savings in archiving and tape mounting costs. However as Bhyat is quick to point out, the greatest benefit is in the company's disaster recovery capability. With the reduced run times, backups can be started early enough to capture everything processed in the preceding 24-hour period instead of just some of the data. Bhyat and the entire MIS staff sleep much better now.

High Performance Access Alternative

IAM is a high-performance indexed access alternative. For use when existing indexed access methods do not meet the performance requirements of the application, IAM will outperform conventional access methods such as VSAM in MVS, XA and CMS environments. For the most part, IAM is transparent to the user and is of little concern to the systems programmer.

VSAM was an attempt to provide an access methodology that gave applications a measure of data independence. Users of VSAM are made to pay dearly for this independence.

An IAM file is a non-VSAM file in the catalog. It is thus devoid of most of the complications inherent to VSAM. It consists of a single MVS dataset with the data, index and file descriptions all contained within the dataset itself. IAM tailors a file's processing dynamically, actually changing buffer processing options during execution. IAM's "Real Time Tuning" is a powerful facility.

Clearly IAM made a significant difference in batch processing at NOREX. In addition, the performance gains provided by IAM have paid off in many subtle ways. As batch performance is no longer a criterion, it has meant a reduced need to rewrite applications. The benefit? Programmers can be put on jobs that more directly benefit the business. NOREX saw corresponding benefits in CICS processing as well.

For example, the use of IAM has made CICS more efficient in two ways. First, in terms of improved system usage, IAM gave NOREX a 25 percent reduction in overall CICS CPU time. Relative to VSAM, it enabled a 45 percent savings in I/O. Second, and perhaps of more significance in the long term, because IAM is self-tuning with dynamic buffering, NOREX's lone CICS systems programmer can concentrate on CICS instead of tuning VSAM.

Finally, the use of IAM translated into significant savings of hardware and related maintenance costs. Under VSAM, NOREX had allocated 12 3380s for VSAM and all 12 were saturated. With IAM installed, only seven were needed, a 60 percent reduction. The other five volumes were freed up for other uses.

At NOREX today, VSAM is used only when IAM cannot be employed, typically for the applications NOREX inherited that still require alternate index files. That situation is rare. Out of 120 files in NOREX's test CICS regions, 106 are IAM files. According to Bhyat, when IAM supports alternate indexes, VSAM files could be retired for good.

Bhyat reports that Version 6 of IAM is virtually 100 percent transparent to users. IDCAMS supports it as a normal VSAM file; no changes are required to JCL. "We've never had to make a program change because of IAM," he says.

If all of these savings sound a little too "easy" to be true, Bhyat agrees. "IAM performs as well as the hardware you give it. If you are running VSE/SP and have a memory-constrained system, overall performance may not be as great as we have experienced. IAM takes advantage of XA storage above the line and probably could benefit even more in an ESA environment."

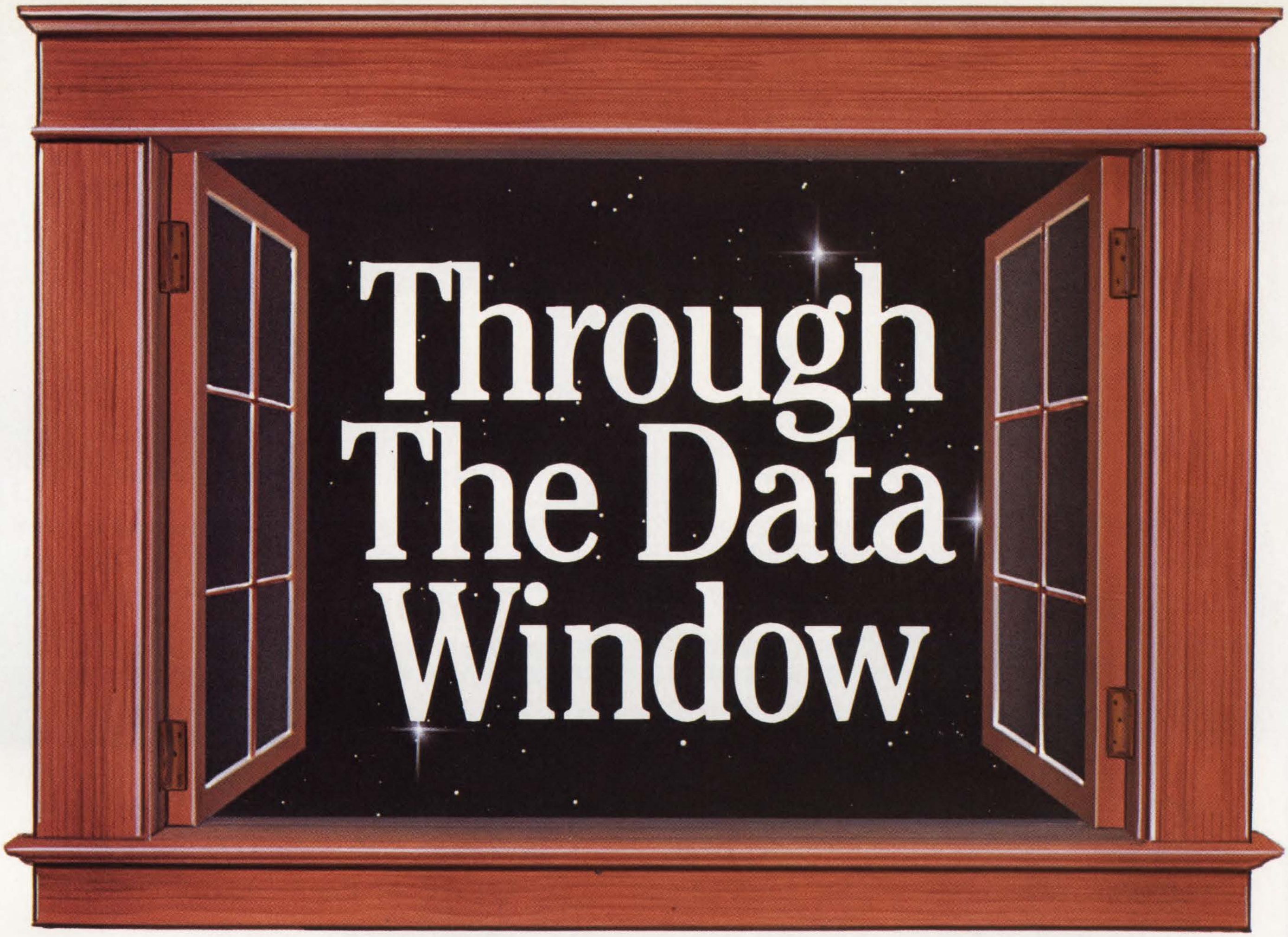
ABOUT THE AUTHOR

John Kador is a free-lance writer concentrating on data processing and high-technology topics.

EDITORIAL EVALUTION

Please circle the appropriate numbers on the Reader Service Card.

1. This article was:
   240 (Interesting/Helpful), 241 (Too Technical), 242 (Too Basic)
2. Would you like more articles on the same subject?
   243 (Yes), 244 (No)



A Look At MVS/ESA's Data Spaces And Hiperspaces

By Michael Haupt

The new MVS/ESA operating system grants programs the use of up to 16 terabytes (TB) or trillion bytes of virtual data. The article, ''ESA Is On The Way,'' that appeared in the January 1989 issue of MAINFRAME JOURNAL showed why the data processing environment of the coming decade mandates vast storage and improved performance. How the ESA/370 architecture expands the existing structure to address this storage while reducing the overhead was the subject of ''ESA's Addressing Architecture'' that appeared in the February 1989 issue.

This article explores ways in which this new storage can be used by application programs. The keys to use of this storage are new ESA facilities, data spaces and hiperspaces.

Data Spaces

In MVS/ESA, programs can refer to data outside of their address space, in a data space. Data spaces are virtual storage structures similar to address spaces. Data spaces enable MVS/ESA to provide application programs with immense data addressability.

Characteristics

Like address spaces, data spaces are variable in size up to a 2GB maximum. There are, however, some significant differences between the two. Data spaces exclusively store data. Instructions cannot be executed in a data space. Also, the entire 2GB area is devoted to data. None of the space is used for common area (nucleus, LPA, CSA, SQA) as with an address space. In addition, data spaces can only be referenced using access registers, a set of 16 registers introduced by ESA/370. Lastly, data spaces do not have address space control blocks and identifiers (ASCB and ASID). They are identified by means of a STOKEN or space token.

Each data space is owned by a single address space. That address space may share the data space with other address spaces.

The data space contains virtual storage. There is no automatic connection with a permanent dataset. When the owning address space terminates, the data space and its contents are lost.

Data spaces are created and deleted by the DSPSERV macro. The RELEASE option of DSPSERV returns all or part of

OVER 1,000 COMPANIES WORLDWIDE ARE BENEFITT

POOL

DASD

Dynamic Data Set Pooling And Standards Enforcement

POOL-DASD

POOL-DASD:

- Makes Pooling Easy
- No EDTGEN's or IPL's to change pools
- No JCL or program changes
- Over 50 fields offered as selection criteria for pooling and standards enforcement
- Sophisticated IF/AND/OR selection capability
- Pre-Allocation standards enforcement
- Installs in 30 minutes
- Provides Management Control
- **NON-SPECIFIC VSAM POOLING**

Customer Comments...

"POOL-DASD has allowed us to go from user-managed storage to system-managed storage and enabled us to position ourselves for the future — **TODAY!"**

- Data Processing Center Manager
Financial Services Company

"Users don't have to worry about finding space for VSAM datasets. POOL-DASD finds it for them **automatically."**

- DASD Administrator
Communications Firm

"POOL-DASD's flexibility and enforcement capabilities returned control of all DASD allocations back to where it belongs — with ME — **I like being in control."**

- Technical Services Manager
Bank

EMPACT software

Corporate Headquarters • 1803 Overlake Dri

NG FROM EMPACT SOFTWARE PRODUCTS INCLUDING:

The Software Product That Stops B37, D37, E37, Abends.

STOP-X37

STOP-X37:

- Intercepts the abend
- Dynamically finds available DASD space
- Allocates the space to the job
- Continues the job without abending
- Produces a detailed audit trail
- Is completely user controlled
- Stops "Primary space not available"
- Dramatically increases DASD utilization
- Helps delay DASD purchases
- No JCL changes
- No program changes
- Installs in 30 minutes

Customer Comments...

"Finally a month-end without X37 abends..."

-MANAGER OF TECHNICAL SERVICES
INSURANCE COMPANY

"In our business, we have to be on time - no excuses! Our "batch window" is very tight. We do not have time to re-run jobs. STOP-X37 makes us look good."

-V.P. OPERATIONS
BANK

"STOP-X37 saved a major job from abending at 3:00 a.m. during the evaluation period...Guess who didn't have to wake up and fix it? I became an instant fan of STOP-X37."

-PRODUCTION SUPPORT MANAGER
MANUFACTURING COMPANY

e • Conyers, Georgia 30208 • (404)483-8852

CIRCLE #119 on Reader Service Card ▲

the data space to its freshly created state.

Private and Shared Data Spaces

Data spaces can be either private or shared. A private data space can be created by any address space. The private data space is used exclusively by its creator. When an address space is swapped, all private data spaces it owns are swapped as well.

Shared data spaces, as the name implies, can be shared between any number of address spaces. Shared data spaces must be created by a non-swappable authorized program (executing either in supervisor state or with PSW key 0-7).

Once a shared data space is created, the owner can allow other address spaces access to the data space. Programs using the shared data space are not required to be authorized.

Data Space Benefits

Data spaces furnish MVS/ESA users with three major benefits: Virtual Storage Constraint Relief (VSCR), increased Reliability, Availability and Serviceability (RAS) and improved performance.

VSCR

Relief from virtual storage constraint is a major benefit of data spaces. Only a portion of a 2GB address space can be used for data. Much of the address space is consumed by program code and the common area. In a data space, on the other hand, the entire 2GB can be dedicated to data.

In addition, an address space can use multiple data spaces. In fact, each address space can access up to 253 data spaces, each 2GB in size. This amounts to 16TB of virtual storage available to each individual address space.

RAS

Data spaces increase data integrity, improving the overall reliability and availability of applications. Each data space contains data from a single file. The data space is isolated from problems both in the application program and in other files. This isolation prevents accidental corruption of the data.

Data spaces also simplify the sharing of data between multiple application programs. Although not currently announced, they offer a potential solution for VSAM's file sharing limitations.

Performance

Data spaces may also improve an application's performance. Because a data space consists of virtual storage, access method I/O may be replaced by paging. While paging is considerably faster, it impacts the entire system.

ESA's performance, especially when using data spaces, will be even more dependent on adequate paging resources than XA.

Hiperspaces

MVS/ESA offers high performance data spaces called hiperspaces. Hiperspaces primarily exist in expanded storage. As data is needed, it must be moved between the hiperspace (in expanded storage) and an address space (in central storage).

The main benefit of hiperspaces is improved performance. Increased use of expanded storage substantially reduces retrieval time over both auxiliary paging and access method I/O.

Characteristics

A hiperspace is a special type of data space. There are, however, a few differences between the two. A hiperspace is backed by expanded storage and, occasionally, auxiliary storage. Also, a hiperspace never resides in central (real) storage even during initialization and hiperspaces cannot be added to an access list. In addition, system code which manages hiperspaces runs in AR mode on behalf of the caller. This permits applications to manage hiperspaces without requiring them to directly use the AR mode and access registers.

Types of Hiperspaces

Two types of hiperspaces provide different performance levels.

Standard or Scroll Type

The standard or scroll type hiperspace resides in expanded storage overflowing into auxiliary storage as necessary. If expanded storage is limited (or not available), standard hiperspaces may exist exclusively in auxiliary storage.

Standard hiperspaces can be created and used by all programs, whether authorized or not.

Expanded Storage Only or Cache Type

The other type of hiperspace, called cache type, resides only in expanded storage and is not backed by auxiliary storage. Since this hiperspace is always in expanded storage, it functions as an extremely high speed disk.

Only authorized programs can use cache type hiperspaces. The owning address is recommended, but not required, to be nonswappable.

Hiperspace Services

Hiperspace Services (HPSERV) control hiperspaces. Programs invoke HPSERV to transfer 4K blocks of data between the hiperspace and an address space. The SREAD/SWRITE functions control scroll type hiperspaces. Cache type hiperspaces are controlled by the CREAD/CWRITE functions.

HPSERV are also created hiperspaces. When creating a hiperspace, the user can specify a number of options which: specify the amount of expanded storage dedicated to the hiperspace; determines whether or not the hiperspace is permitted to overflow into auxiliary storage; and select hiperspace access option: read only, read/write or destructive read.

Performance Benefits

The main benefit of hiperspaces is improved performance. Increased use of expanded storage substantially reduces retrieval time over both auxiliary paging and access method I/O. Additional ESA enhancements give added performance boosts for hiperspaces.

For example, when moving data within a hiperspace, the Real Storage Manager (RSM) that controls hiperspaces in expanded storage can frequently complete the action by updating control blocks rather than performing a physical move.

Likewise, moving 4K of data from a data space to an address space casts out 8K of processor cache (high speed buffer). Hiperspace's read/write operations, on the other hand, do not destroy processor cache.

Summary of the Spaces

Figure 1 shows ESA supports or backs virtual storage for address, data and hiperspaces.

Both address and data spaces can be backed by all three forms. Normal paging operations control whether a specific page is in central, expanded or auxiliary storage.

The difference between address and data spaces lies in their function. Data spaces are functionally limited to storing data. Address spaces provide for program execution as well as data storage.

Neither type hiperspace ever exists in central storage. A program must use HPSERV to move the data into the address space's central storage before it can be used.

Auxiliary storage backs the scroll type hiperspace if its contents are greater than the amount of expanded storage dedicated to the hiperspace. Cache type hiperspaces, on the other hand, are only backed by expanded storage.

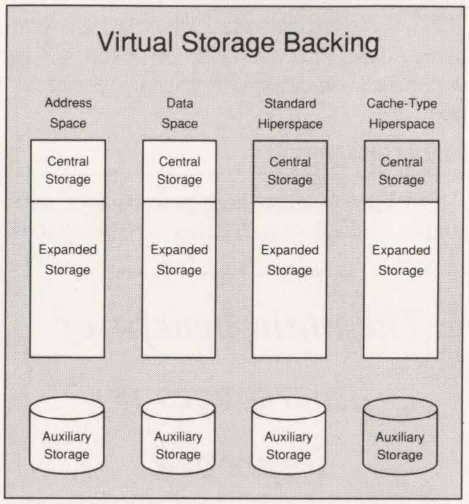
Application Use

Now that ESA's new space facilities are understood, how applications programs can take advantages of them will be examined.

AR Mode

The new AR addressing mode, described in ''ESA's Addressing Architecture,'' is, by far, the most powerful way for applications to use data spaces.

FIGURE 1



Using AR mode, an application can move a field from one data space into another. Or the program can add two values, each found in different data spaces, storing the result in a third.

All this activity is done directly in the data spaces without first moving the data into the address space. This makes data spaces ideal for large tables, matrices and other control structures. Each program could have its own private copy of a standard table, to modify at will. Or, by writing a simple control program to create the data space, the installation could share a single copy of the table between all users.

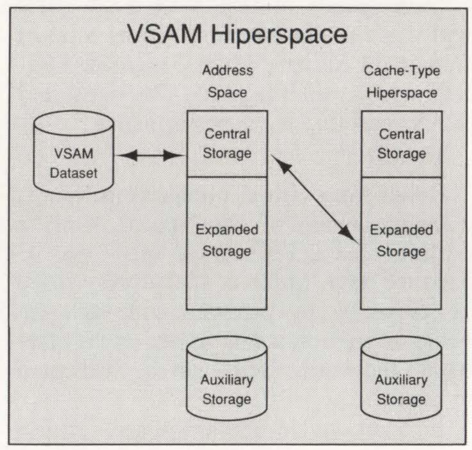
The biggest limitation with the AR mode is that it is only available to Assembler language programs. This will limit its use, in the near future, for most in-house developed applications. For non-Assembler programs, the installation could write a small data space server to move data between the data and address space.

The use of AR mode will doubtless be incorporated, over time, into IBM's strategic large system software. The ''standard environment'' will probably be an address space containing the processing programs, shared data spaces for common data and a private data space for each user.

Buffering

While applications will take some time to directly use AR mode, they can gain more immediate benefits for the use of buffer pools to reduce I/O. One technique for doing this is the VSAM hiperspace, shown in Figure 2.

FIGURE 2



Authorized applications can establish a cache-type hiperspace to store VSAM buffers. The service, BLDVRP, builds LSR buffer pools. BLDVRP indicates the number of buffers both in the address space and in a hiperspace.

The program processes data in central storage buffers. Prior to reading from the disk, VSAM does a lookaside trying to locate the needed Control Interval (CI) in an existing buffer in either central or expanded storage. If a hiperspace buffer contains the CI, it is moved into an address space buffer in central storage before processing. A least-recently-used algorithm maintains the contents of buffers in both the address space and the hiperspace.

Normally, the address space should contain only a few buffers, likely to be highly active and remain in central storage. When inactive, address space buffers are subject to normal paging into extended or auxiliary storage.

Data-In-Virtual

Data-In-Virtual (DIV) is a facility introduced in MVS/SP 2.2. DIV uses the MVS paging algorithm to provide high performance for large datasets.

Early DIV

DIV uses a VSAM file organization called Linear Dataset (LDS). CIs in the LDS are always 4K. Unlike other VSAM organizations, like KSDS, CIs in the LDS contain data in the exact form that it is used by the application. There is no VSAM control information or free space within the CI.

DIV enables applications to map a portion of virtual storage to the LDS. Once mapped, the paging mechanism handles all references to those virtual addresses. Page faults cause data to be retrieved from the LDS. Unused addresses may never be brought into storage. Only CIs that are changed are rewritten to the LDS.

DIV is primarily used by graphics and engineering packages. The LDS would contain a drawing. An engineer normally retrieves a drawing to work on one or two small sections. Without DIV, there were two choices: use the normal VSAM I/O routines or read the entire drawing into virtual storage. The first choice is much too slow for interactive terminal work. The second choice is fast while working but necessitates long delays while the entire drawing is loaded or saved.

DIV provides a workable solution. I/O through the paging mechanism is considerably faster than for other VSAM file organizations. Because data is only read when needed, times to prime a dataset are drastically reduced. Saving options permit the user to write changed CIs upon command or to discard any changes not previously saved.

DIV's solution reached a limited audience. It is supported directly in VS FORTRAN Version 2 and used by a few other products like CATIA and EDS/GM.

DIV in ESA

ESA incorporates a number of enhancements into DIV. One enhancement is an optional sequential support. Often, the use of data is clustered. Filling the screen with a portion of a drawing may require data from five to 10 pages. Presently, each page requires a separate I/O. An option with ESA retrieves the referenced page plus up to 15 additional pages. Thus, the application can retrieve up to 64K with each page fault. Only changed pages are written, however, no matter how many are sequentially read.

Another improvement increases the shared use of data objects. Currently DIV objects permit either a single write or multiple read activity. New options on the DIV macro permit multiple users to obtain a private local view of a shared DIV object. Each user can write his private local view. DIV handles serialization.

The final ESA enhancement enables DIV to map to data spaces and standard hiperspaces. This gives DIV users the ability to use much larger objects as well as increased performance when using the hiperspace.

Data spaces and hiperspaces are new storage structures that provide applications with expanded virtual storage, improved reliability and increased performance.

Data Windowing

Data windowing is an extension to DIV. Applications can use data windowing to map either temporary or permanent data objects.

Data windowing services, listed in Table 1, are implemented using a CALL interface. Programs can be written in COBOL, FORTRAN, PL/I, Pascal, as well as Assembler. Data windowing is available whether the program is in 24-bit or 31-bit addressing mode.

TABLE 1

Data Window Services

Name	Description
CSRIDAC	Identify and access a linear object; either permanent or temporary; option to create a new object
CSRVIEW	Establish a view of a linear object in virtual storage
CSRSCOT	Scroll out to capture the current view
CSRSAVE	Commit all changes in current and scrolled views
CSRREFR	Refresh current and scrolled views discarding all changes since previous save

Application programs establish a user defined area as a window into the data. The program can reference the data directly or scroll, moving the window forward or backward through the data.

Data windowing functions with two types of data objects, permanent and temporary. Permanent data objects are VSAM linear datasets. As with DIV, data windowing provides applications improved performance for large datasets. Physical updates can optionally be deferred until requested or program termination.

Temporary data objects are standard type hiperspaces. Data windowing handles moving data between the hiperspace and the address space. With temporary objects, data windowing permits multiple hiperspaces to be seamlessly concatenated. The application appears to have a data object up to 16TB.

Conclusion

MVS/ESA introduces us to data spaces and hiperspaces. These new storage structures provide applications with expanded virtual storage, improved reliability and increased performance.

Application programs have a number of options to use data spaces and hiperspaces. High volume applications requiring high performance are the ones which will benefit most from these new features.

Few installations will restructure their in-house developed applications in the near future. The next article will examine how MVS/ESA internally uses these new features to provide performance benefits without touching applications.

ABOUT THE AUTHOR

Michael Haupt, CDP, is a senior associate with Austin Consulting in Rosemont, IL. During his 15 years in data processing, he has worked extensively with large scale IBM systems and has been active in professional associations and user groups. Austin Consulting, 9801 W. Higgins Rd., Rosemont, IL 60018, (312) 696-0500.

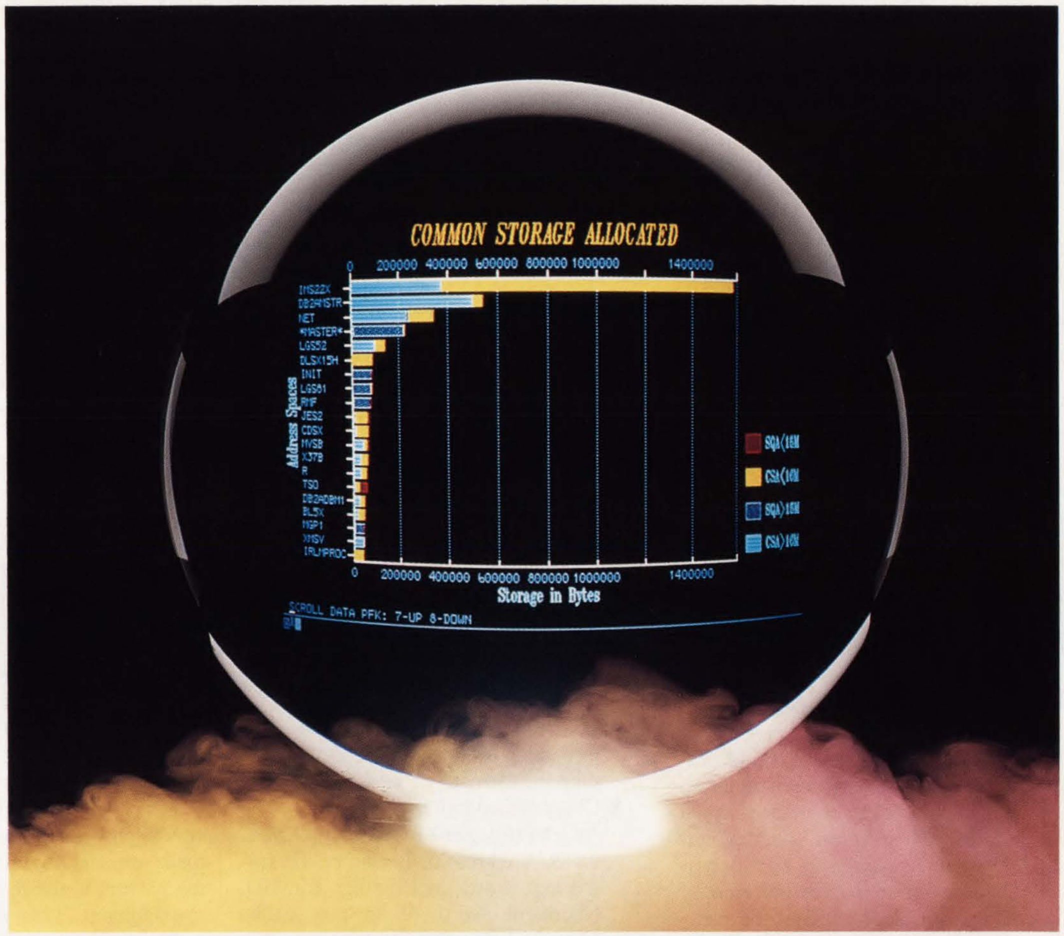
EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
245 (Interesting/Helpful), 246 (Too Technical), 247 (Too Basic)

2.) Would you like more articles on the same subject?
248 (Yes), 249 (No)

INTRODUCING THE FIRST REAL-TIME MONITOR FOR CSA.



Introducing the new Common Storage Monitor for RESOLVE PLUS, the first in-depth tool for monitoring users of CSA and SQA on MVS/XA. And the most effective way to reduce time consuming, and costly, IPLs caused by common storage constraints.

Control CSA creep.
Common Storage Monitor helps you control "CSA creep", the slow, hidden increase in common storage allocation that can ultimately result in system degradation and even failure. It is the only monitor that allows you to account for common storage usage by individual user. So you can identify applications that are abusing common storage and recover wasted CSA held by terminated tasks.

Pinpoint CSA usage.
Common Storage Monitor displays allocated storage for each address space by job name. Operating in either ISPF or command mode, you can display as much detail as you need to identify the user responsible for the allocation, and to analyze how the storage is being used. Password-protected RESOLVE PLUS services are also provided to free areas of CSA once they have been identified. Online color charts give you an easy-to-interpret overview of common storage allocation.

For more information on the Common Storage Monitor for RESOLVE PLUS Version 3.0.0, call Marty Johnson today. In California: 800-624-5566. Outside California: 800-822-6653. Boole & Babbage, Inc. 510 Oakmead Parkway, Sunnyvale, California 94086.

Boole & Babbage

International sales and support provided through The European Software Company and a worldwide distribution network.

NOW AVAILABLE IN A STANDALONE VERSION!

CIRCLE #160 on Reader Service Card ▲

PRODUCT REVIEW

The Information Systems Series

At Last — Generic Standards And Procedures!

By Judy Glick-Smith

Wouldn't it be wonderful if you could get customized standards and procedures for your data processing shop without paying an arm and a leg in dollars and staff hours. The Information Systems Series (ISS), developed and recently released by the Technical Publications Division of the Computer Resources Group, Inc. (CRG), is a generic standards and procedures product with specific instructions for customizing with your own data center information.

CRG has been providing programming, systems support and consulting services to the data processing industry for more than fifteen years. ISS is a direct offshoot of this experience base.

Who Needs ISS

I have never been in a data processing shop that did not need standards and procedures, yet when the shop has standards, keeping them current is a chore done when time permits. Most often, documentation, if it exists, is scattered and incomplete. Data center managers are usually aware of this problem but find the prospect of developing and maintaining current and complete documentation overwhelming. If you are in this position — not being able to see the forest for the trees — then ISS is for you.

Case in point: William Wimberly is manager of systems operations for Dalmo Victor, a division of General Instruments. His data center is a 20-year-old RJE shop. One year ago it was full of spaghetti code and no documentation. Not wanting to reinvent the wheel, Wimberly began a search for a set of standards and procedures from which to use as a basis for his own shop. Standards from other shops just did not fit his needs. When he called CRG, ISS was in the alpha stage. He was so impressed with ISS, he bought the product when it was still in the beta stage. "It is a well-organized shell that only needed minor modifications to fit our shops," he reports, adding he would recommend ISS to anyone needing a boost in getting started writing standards and procedures.

The Product

ISS is a documentation framework for developing customized standards and procedures specifically for VM shops. However, more than an outline, ISS details a fool-proof methodology for developing and maintaining current documentation.

You need the following resources to install and maintain ISS:

- Someone to designate as the series coordinator
- An IBM (or fully compatible) personal computer
- A hard disk and a 5¼″ diskette drive (or two diskette drives)
- A word processing package such as WordPerfect or MultiMate
- Enough memory to run the word processing package (usually 256K)
- A printer that prints upper and lower case letters.

The product arrives with the following items:

- Four *Information Series Guides* in three-ring binders, printed in the untailored format
- Diskettes containing the text of the *Information Series Guides* in ASCII or DCA format or in one of the following word processing formats: Microsoft Word, WordStar, MultiMate and WordPerfect
- The *Instruction Booklet* to help you convert the supplied materials into a set of customized standards and procedures for your data center.

The ISS *Policy Guide* is designed for use by data processing managers and includes objectives, development methodologies, security information, roles and relationships and other general information necessary to the smooth operation of a data processing shop. This manual also discusses external standards such as IBM's Systems Application Architecture (SAA). Having this information in one place helps keep the entire data center in tune with the same goals and guidelines.

The *Data Center User Guide* contains information everyone, including technical and non-technical personnel, needs in order to effectively use the data center. Information contained in this manual includes operating hours of the data center, hardware and software configuration, cost of services, a description of user support, how to use a terminal, an introduction to CMS and instructions on how to use it, how to protect data and available software. Appendices include forms used in the data center and a contact list.

This is a reference manual designed to minimize the time needed to find out answers to common questions and to minimize the cost in teaching personnel about the data center.

The *Systems And Programming Standards* manual is designed for use by data processing staff. This manual contains information such as the data center's system development life cycle methodology, system development practices (design and development procedures), change and problem management procedures, naming conventions, standards for database management, language standards and documentation standards.

Once these systems and programming standards are in place and if they are enforced, productivity and efficiency will increase through reduced time for maintenance, training new personnel and cross-training existing staff.

The *Operations Guide* is targeted at operations and technical support personnel and includes such topics as emergency procedures, change and problem management, how to shut down and IPL the system, network management, DASD management, tape management, printer operation and maintenance, physical security, production control and scheduling, performance monitoring and disaster recovery procedures. This manual is designed to facilitate the successful operation of the data center.

The *Instruction Booklet,* directed at upper management and the series coordinator (the person assigned to coordinate the development of ISS), contains the methodology for creating and maintaining ISS. The first part of the booklet is directed at management and outlines the steps that need to be taken to publish ISS. The rest of the booklet details the steps the series coordinator should take to create and maintain the customized data center standards and procedures.

These steps include:

- Preparing for the development of ISS such as defining the review process and introducing ISS to the data center
- Customizing each section, even going so far as to suggest which chapters of which manuals should be developed first, second and so on
- Printing and issuing the final product.

If you carefully follow the steps detailed here, you will have a quality product that your staff will quickly come to appreciate.

Implementation

If you follow the *Instruction Booklet,* implementation is a breeze. It stresses repeatedly the importance of upper management support. Upper management assigns the series coordinator and introduces the ISS project to the rest of the data center.

The series coordinator, following the *Instruction Booklet,* plans the implementation and loads the floppy disks in the word processing system. The coordinator makes the first modifications to the title pages and the headers and footers of each manual. Then, using available resources within the data center, the coordinator develops each individual section in the ISS.

Within each manual, where shop-specific information needs to be included, panels of asterisks enclose descriptions of information to be added, questions to be answered and other comments. For example, in the *Data Center User Guide,* in the section 6.5, ''Moving Data between Machines,'' there is the following panel:

> ''Possibly more important than the procedures themselves, is the issue of security. Data that is adequately secured on the mainframe can be much more vulnerable on a personal computer and even more when copied to diskettes. Discuss here whatever measures your installation feels are necessary to protect data of varying sensitivity.''

This panel defines the content of this section. The series coordinator removes the panel after researching and adding the appropriate information. The system automatically repaginates the document. The coordinator can easily locate these notes using a global search so that all questions, decisions and additions can be resolved before final printing.

The coordinator is also responsible for seeing that the review process is thorough and timely in order to assure a correct and current final product.

Limitations

The ISS is in hard copy manual format. The only person who can access information contained in the manuals on-line is the series coordinator. The problem with paper based manuals is often they are out-of-date the day they are distributed because of the lead time required for printing.

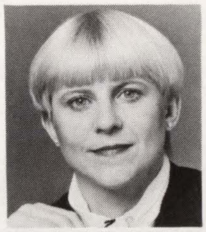
Also, the product is only available for VM shops. However, an MVS version will be released in the spring of 1989, according to the vendor. Those of you who run VSE shops may still want to consider ISS. Repalce those sections specific to the operating system with your own operating system information. CRG has already organized the information for you. All you have to do is fill in the blanks.

Conclusion

The ISS is an organzied, well written document on which you can build your own customized data center standards and procedures. It was designed and written by experienced data processing professionals who enjoy writing documentation. Information is presented logically and procedures are in step format (1., 2., . . . and so on). By following the detailed *Instruction Booklet,* you can be sure the end product will be a complete, usuable document that you did not know how you did without for so long.

For more information, contact Technical Publications Division, The Computer Resources Group, Inc., 303 Sacramento Street, San Francisco, CA 94111, (415) 398-3535.

ABOUT THE AUTHOR



Judy Glick-Smith owns Integrated Documentation, a company specializing in publishing internal documentation for data processing shops. Integrated Documentation, 2995 LBJ Freeway, Suite 200, Dallas, TX 75234, (214) 888-1003.

M•V•S performance Notebook

Tuning parameter enhances IBM mainframe performance.

By Mark Friedman

The CPENABLE option enhances the performance of an IBM mainframe. It is an infrequently used and little understood tuning parameter in the OPT member of SYS1.PARMLIB.

In MVS/XA, only one processor of a multi-engine CPU is initially enabled for interrupts. Whenever an I/O completes, the channel subsystem signals the processor via an interrupt. The idea of concentrating all interrupt processing on one processor is to improve the performance of interrupt processing and limit the impact of interference from interrupt processing on the other processors.

The Interrupt Handler (IOS)

Servicing interrupts is one of the highest priority tasks of the system and the first step in interrupt processing is to disable the processor for further interrupts. Interrupt processing cannot be interrupted! Then the status of the external event causing the interrupt is evaluated and processed. If the I/O has completed successfully, the interrupt handler (IOS) finds the task associated with the I/O request and readies it for execution.

Next, IOS checks for requests queued to the device that generated the interrupt. Since the device is now available (ignoring for the moment any shared DASD considerations), IOS starts the queued I/O request.

Finally, IOS issues a Test Pending Interrupt (TPI) instruction to see if there are interrupts delayed in the channel subsystem while the disabled system was processing the current request. If there is a pending interrupt, IOS accepts it and begins processing it. The entire process continues until there are no more interrupts pending. At this point IOS terminates and re-enables the processor for interrupts.

The delay experienced by I/O events that attempt to interrupt a disabled processor can be avoided in a multi-processor environment. One of the architectural enhancements provided with MVS/XA was that any processor could process an interrupt from any device, via *floating channels* managed by an outboard I/O processor.

Having all processors enabled for interrupt processing at all times sounds good in theory. However, in practice this has a very negative impact on performance. The reason is that IOS code running at a high priority will pre-empt application code.

As IOS begins execution, IOS code and data areas are fetched into the CPU's cache memory. The IOS code and data segments flush out the application code that was already in the cache buffer. When interrupt processing is over and the original application can resume, its frequently used code and data segments must be reloaded into cache. Both IOS and the interrupted application experience high cache buffer miss rates.

The result of frequent *context switching* (IBM's term) is high cache miss ratios that result in seriously degraded performance. CPU cache performance is critical to the overall performance of an IBM mainframe. Memory references that are resolved from cache are several

ANNOUNCING

A New More Productive Era in Systems Programming

The Systems Programming Environment
Only in the SAS/C® Compiler

Until now, higher-level languages just couldn't hack it in the systems programming world. Too many issues stood in the way—inefficiency, poor access to low-level system services, bulky and intrusive library requirements, and inflexibility in addressing the IBM 370 architecture.

But now you can write systems-level routines faster and maintain them better than you ever could with assembler—with the SAS/C compiler's exclusive Systems Programming Environment (SPE).

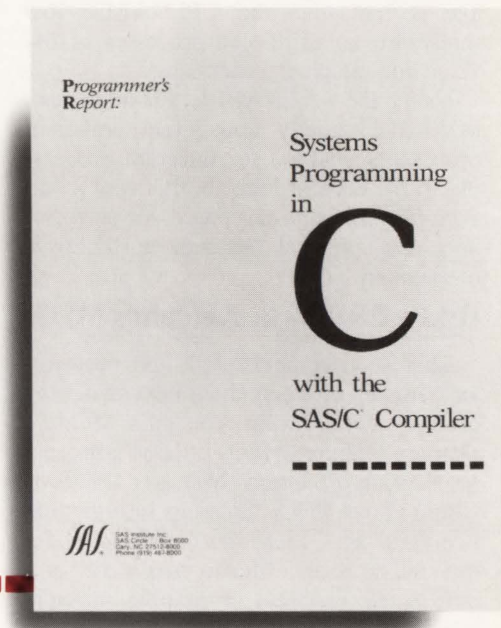
SPE is an extension to the C language that greatly simplifies the coding of user exits, tools, and utilities for JES2, VTAM, CICS, TSO, GCS, and other systems software. Included are support routines that allow you to write and execute C programs and a compact runtime library that features both general purpose and system specific functions for memory management, interrupt handling, low-level I/O, and more. There's also a utility that translates assembler DSECTs into C structure definitions—an enormous time-saver when you're writing programs that interface with assembler.

Together these tools provide a freestanding C environment designed to interact with the operating system the same way assembly language programs do. With SPE, your C programs can:

- call existing assembler routines
- generate any machine instruction in-line
- easily access system data and control blocks
- exploit BSAM or CMS file system I/O
- process asynchronous events and interrupts
- directly use SVCs and DIAGNOSEs

Then, at compile time, the SAS/C compiler's global optimizer will compress your code to produce routines that rival assembler for speed and efficiency.

With frequent updates and knowledgeable technical support—both provided free—the SAS/C compiler is the best investment you can make toward greater systems programming productivity.



Learn More in a FREE Programmer's Report

To find out more about systems programming with the SAS/C compiler, simply clip the coupon below and mail today. We'll send you our new **Programmer's Report: "Systems Programming in C"**. Or call us today to find out how you can receive the SAS/C compiler for a free, 30-day evaluation.

SAS Institute Inc.
SAS Circle □ Box 8000
Cary, NC 27512-8000
Phone (919) 467-8000
In Canada, call (416) 443-9811

The SAS/C compiler runs under MVS (370, XA, and ESA) and VM/CMS on IBM 370/30XX/43XX/937X, and compatible machines.

____ **Yes, send me a FREE copy of "Systems Programming in C".**

____ **Contact me with details of a FREE 30-day trial of the SAS/C® compiler.**

Mail to: SAS Institute Inc., Attn: ME, SAS Circle, Box 8000, Cary, NC, USA, 27512-8000. Telephone (919) 467-8000.

Please complete or attach your business card.

Name ____________

Title ____________

Company ____________

Address ____________

City ________ **State** ____ **ZIP** ______

Hardware System ________ **Operating System** ________ times faster than those that access main memory.

CPU cache hit ratios of 95 percent plus are not unrealistic, once a running process has a bit of a chance to acquire its *working set* of instructions and data areas in the cache. With frequent context switching, each running process essentially gets a *cold start* from cache since its working set has been flushed by the previous user.

Enabling as few CPUs as possible for interrupt processing is a software technique that controls the amount of context switching in the hardware. Of course, since the rate of I/O interrupts fluctuates, it is difficult to determine precisely what number of CPUs to enable. If too many processors are enabled, overall performance is reduced due to frequent context-switching. If too few, too many I/O interrupts are delayed in the channel subsystem.

MVS calculates the percentage of interrupts that are processed via TPI. If the percentage of interrupts handled via TPI is greater than the CPENABLE high threshold, another processor is enabled for interrupt processing. If the TPI percentage is less than the CPENABLE low threshold, an additional processor is disabled for interrupt processing.

Using the CPENABLE threshold test, MVS dynamically adjusts the number of processors enabled for interrupt processing to the current load. In this way a balance between optimal processor performance and minimal I/O interrupt delay is maintained.

IBM's PR/SM & Amdahl's MDF

This section deals with the performance impact of frequent context switching by a *hypervisor* like Amdahl's Multiple Domain Feature (MDF) or IBM's Processor Resource/Systems Manager (PR/SM) sitting above MVS. Since its introduction two years ago, MDF has had a successful run. MDF is an efficient hardware solution to the problem of running multiple copies of an operating system on a single processor without compromising reliability and system integrity. Using MDF, you can carve up an Amdahl processor into multiple systems of an arbitrary size. This is called *logical partitioning* of the processor.

The MDF hardware and firmware act as a hypervisor, sharing the processor logically among two or more copies of an operating system. A logical partition for each operating system is established that can be set up for either dedicated or shared access to one or more CPUs. Memory and channels must be dedicated to a single partition or domain. Unlike physically partitioned processors, under MDF the configuration can be broken into asymmetric logical partitions. There are also options to change the configuration on the fly.

Processor sharing is implemented through time-slicing but if one domain is using less than its logical share, the processor can be made available to other partitions. Because the time-slicing is implemented on dedicated hardware, the scheme is efficient. The major source of overhead in running MDF results from the additional context switching that results from switching back and forth between partitions. The more partitions, the greater the overhead.

One popular use for MDF is to allocate a small partition to run an MVS test system on the same machine that is running the production workload. System programmers working the graveyard shift to bring up a new MVS can appreciate the utility of a feature like this. Wives and other loved ones appreciate it, too. So do data center managers who can now have their top systems jocks available during the day-to-fix problems.

Originally positioned against an IBM software alternative, MDF proved less expensive than running multiple copies of MVS under VM/XA (or VM/SF). Amdahl's licenses for MDF were designed to be competitive with VM/XA but MDF proved to be far more efficient to run. While VM/XA could devour anywhere from 20 to 40 percent of a large processor, MDF overhead is in the two to 10 percent range. Implementing MDF is a breeze; however, running VM/XA supporting multiple MVS guests means acquiring serious VM expertise. Basically, it was no contest.

The only sticky part was having an Amdahl processor to run MDF. Amdahl's revenue growth over the last two years suggests that the utility of MDF on Amdahl's behemoth processors was overcoming a lot of objections to breaking from the IBM fold.

One interesting aspect of MDF is that logical partitioning provides additional configuration flexibility that is valuable for large processors. The test partition that you carve out this month can be joined to one of the production partitions next month. A 100-MIPs machine can be shared among two, three or more workloads that do not otherwise fit neatly or economically into smaller, individual machines.

The success of MDF prompted IBM to introduce a comparable feature for its 3090 series last year called Processor Resource/System Manager (PR/SM). Given that Amdahl pulled in almost two billion dollars in revenue in 1988, the most successful year yet for any IBM mainframe plug-compatible vendor, IBM's rapid response to MDF should not be too surprising.

Most observers would say that MDF and PR/SM are similar, and they are, except for one important feature — the way external interrupts for an inactive partition are handled. In MDF, interrupt processing is delayed in the channel subsystem until the partition is dispatched. In PR/SM an interrupt for a higher priority partition will be serviced immediately, preempting the currently dispatched partition.

Which is the better approach? The correct answer is the usual *caveat:* "It depends." There is really no wrong or right way to handle interrupts for a dormant domain, only trade-offs. Once the trade-offs in the different approaches are understood, it will be clear how the CPENABLE option will help to control the downside risks of running a favored workload as a logical partition under either MDF or PR/SM.

The next article will illustrate how CPENABLE can figure into evaluating and tuning a logically partitioned processor running under, depending on your machine vendor, either Amdahl's MDF or IBM's PR/SM.

ABOUT THE AUTHOR

Mark Friedman is a director in the Technical Division of Landmark Systems in Vienna, VA. He is a frequent speaker on MVS performance topics at SHARE and CMG meetings. Friedman holds a Master's degree in Computer Science and has 10 years experience as an MVS performance analyst and a developer of performance analysis software tools.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
250 (Interesting/Helpful), 251 (Too Technical), 252 (Too Basic)

2.) Would you like more articles on the same subject?
253 (Yes), 254 (No)

Remote printing on a PC without a single compromise.

Until now you've had to rely on a S/36 or AS/400® to deliver remote printing. Now a PC with BARR software and adapter sustains print speeds of 6,000 lines-per-minute and line speeds of 128,000 bits-per-second. Only BARR maintains all this performance with the reliability and ease of use PC users expect.

BARR/SNA RJE or BARR/HASP software drives up to five printers from a single PC. What's more, you can enter data, print, and receive output all simultaneously – without interruption. BARR's advanced multi-tasking software easily manages even the most complicated tasks, including LAN access, tape support, file transfer, and special forms printing. In addition, BARR offers one year of friendly, dedicated customer support with each purchase.

Communications adapters and software are available for the IBM PC, PS/2, and compatibles.

Try BARR for 30 days. We've helped thousands save millions. Call 800-BARR-SYS.

Protocols		
SNA RJE with Multiple Sessions		
HASP – BSC Multileaving		
Host Systems		
JES2	DOS/POWER	CDC NOS
JES3	RSCS	VS1/RES

BARR

BARR SYSTEMS INC.
2830 NW 41 Street, Gainesville, FL 32606
800-BARR-SYS, 904-371-3050, FAX: 904-371-3018

AS/400 is a trademark of IBM Corp.

CIRCLE #91 on Reader Service Card ▲

Large DB2 Buffer Pools

May Hurt On-line Transaction Performance

By Joel S. Goldstein

A general guideline for DB2 performance is to make the buffer pool (usually BP0) as large as possible within the limits of available real and expanded storage (paging to expanded storage is all right but paging of the buffer pool to DASD will hurt performance and must be avoided). Since DB2 does not have to sequentially search the pool to find the desired pages, there is little overhead involved and this approach will eliminate I/O for frequently referenced index and data pages.

There have been several studies at large manufacturing companies that have reduced the elapsed time for large table scans by almost 50 percent; the CPU time was reduced by more than one hour. This was accomplished by increasing the number of buffers to 4,000 from a starting point of several hundred. Since the resulting buffer pool (alone) is 16MB, real and expanded memory were not a problem for this user. I have heard that one company is using 17,000 buffers so that large portions of the frequently referenced tables remain resident in the buffer pool.

All That Glitters Is Not Gold . . .

A recent experience indicates that the decision to allocate a large buffer pool must be evaluated more closely, based on the *on-line* transaction profile and function. The question is, "What pages stay in the pool, when are updated pages written back to the databases and what else is affected by the size of the buffer pool?"

Frequently referenced pages that are not updated will remain in the pool (unless they get forced out by a large tablespace scan). Pages that have been updated are written (forced) to the log at commit points but will remain in the buffer pool for reuse by another function or transaction. The actual rewrite of the updated page into its table occurs asynchronously. This is somewhat of an over-simplification but adequate for present needs.

Most importantly, for purposes of this discussion, there is a vital system function that is affected by the size of the buffer pool, sequential prefetch. Ignore the bottom end of the scale. When the number of specified buffers is between 224 and 999, the prefetch quantity is 16. When the number is 1,000 or more, the prefetch quantity becomes 32.

Take a quick look at sequential prefetch. What is it? Prefetch is, just as it

with DB2!

PLATINUM *technology, inc.*
555 WatersEdge Drive
Lombard, IL 60148-9930
(312) 620-5000 FAX (312) 953-1923

For further information, in-house demonstration, or our exclusive no-obligation product evaluation call:

1-800-442-6861

IBM Business Partner

CIRCLE #7 on Reader Service Card ▲

DB2 Performance Advisor

Q *I am experiencing extended elapsed times in DB2. The elapsed times are more than three seconds but the CPU time is less than .1 seconds. The transactions are only causing three to five I/Os, less than 10 getpages and use static SQL. This is only occurring for a small set of inquiry transactions. What can be causing this when the rest of the system provides much better response?*

A You may have been bitten by the DB2 "defaults." Quite specifically, Closerule and/or Validate. The default for Closerule when defining a Table or Index is YES. Thus, DB2 must open and close the table/index each time they are accessed by a Plan. It is also quite possible that the Table was specified properly (Closerule=N) but the specification was overlooked for the Index. Another item to check is the Validate option used to bind the Plan. If Validate(BIND) was not specified, the Plan is being Re-Bound each time it is executed. Try a Select * from SYSTABLESPACE where Closerule = 'Y', also query SYSPLAN where Validate = 'R'. For both of these queries, you should also qualify on Creator and/or Name to reduce the output.

If this does not address the problem, the next topic is resource contention. Since your problem statement implied that you have a performance tool, the next area to investigate is locking contention by another user of application. It does not sound like there is a DASD problem since other transactions provide better response and utilization profiles.

Q *What are the performance implications of variable length columns? The on-line transactions that reference the table will be mostly inquiry but we are expecting about 10 percent of the transactions to perform updates. At the present time, there are no plans for insert/delete transactions. The table will contain about two million rows and the column in question would vary from 70 to 210 characters. The application will process approximately 9,500 transactions per hour during peak load periods.*

A Excluding some additional complexities at the application programming level, the two considerations are data access and logging of updated rows.

Since you did not indicate whether the variable length column will be updated, look at several possibilities. If the variable column is updated and its length changes, DB2 will log the entire row. Additionally, if the new length is too long for the row to fit on the page, it will be moved to a new page. The IBM recommendation is to place variable-length columns at the end of a row. There is a good reason for this guideline. If a fixed-length column was used in a select and the fixed column was positioned after the variable column, DB2 would have to calculate the position of the column before it could be retrieved or compared to a predicate. This would occur for every row and would add significant overhead to the system.

DB2 logs an updated row from the column that has been updated through the end of the row (excluding the case of a variable column becoming longer). If the variable column will usually be the update candidate, and extended, there may be an impact upon the logging function. Fixed columns that will be updated should be positioned near the end of a row, if possible.

Also, have you analyzed the expected distribution of lengths for the variable column? Depending on the average length and DASD saving, you will have to determine whether the performance cost and additional programming justify the variable length column.

Editor's Note: *Readers with questions concerning DB2 performance are encouraged to submit them using the Reader Service Card. Joel Goldstein will answer selected questions in each issue of MAINFRAME JOURNAL.*

sounds, the reading of pages of data into the buffer pool before they are actually requested. The buffer pool is "primed" with the data before the *getpage* is issued. The omniscient black box, the *Optimizer,* determines if prefetch will be used for the query. This is based upon the number of pages that may have to be scanned to satisfy the query. Assume a positive decision is reached and the prefetch flag is on.

16-Page Sequential Prefetch

Using 16-page prefetch, the following scenario occurs:

- Read the first 16 pages synchronously – 16 I/Os
- Read the next 16 pages synchronously – 16 I/Os
- As the getpage for page 17 is issued, do a prefetch read for pages 33 through 48 – 1 I/O
- As the getpage for page 33 is issued, do a prefetch read for pages 49 through 64 – 1 I/O
- And so on.

Sequential prefetch will only be used if the Optimizer has determined that the query will scan 40 or more pages.

32-Page Sequential Prefetch

Using 32-page prefetch, the scenario will be similar to the preceding one, except that the increments or trigger points have doubled:

- Read the first 32 pages synchronously – 32 I/Os
- Read the next 32 pages synchronously – 32 I/Os
- As the getpage for page 33 is issued, do a prefetch read for pages 65 through 96 – 1 I/O
- As the getpage for page 65 is issued, do a prefetch read for pages 97 through 128 – 1 I/O
- And so on.

Sequential prefetch will only be used if the Optimizer has determined that the query will scan 80 or more pages. Note that *everything* has doubled along with the prefetch quantity.

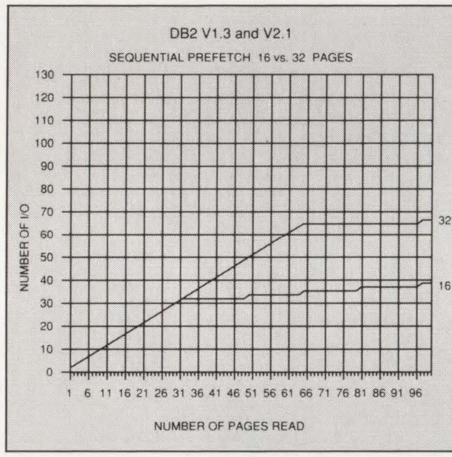
This Hurts . . .

If the on-line environment includes many transactions that scan small but

mostly unique areas of many different tables, the effect upon throughput and response time will be quite noticeable. Unless more than 1,000 sequential pages of a table will be scanned, 32-page prefetch will seriously degrade performance and response time.

You want large buffer pools to eliminate I/O. Prefetch was created to reduce the number of I/Os necessary to retrieve large answer sets and to reduce (hopefully eliminate) any wait time for I/O comple-

F I G U R E 1



tion. We are really addressing an application design question: frequency of data reference. The corresponding question is, is the cost of system resources (in this case, storage) necessary to support buffer pools of 60 or more megabytes.

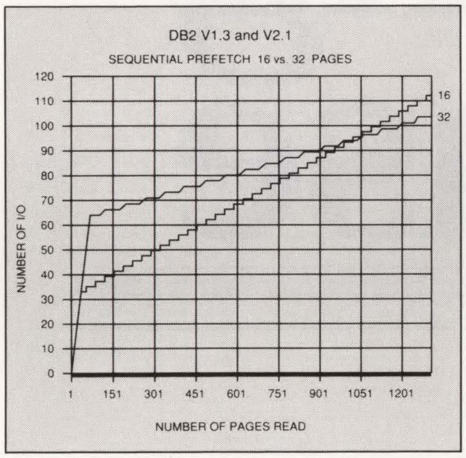
Figures 1 and 2 illustrate the additional I/Os required when 32-page prefetch is used and the data is not in the buffer pool. The graph in Figure 1 is a subset of the second and accentuates the fact that the number of I/Os required to scan 60 to 100 pages almost *doubles* when the prefetch quantity is 32. The graph in Figure 2 illustrates that the larger prefetch quantity will not provide any benefit unless significantly *more* than 1,000 pages will be scanned.

Presently, the only possible solution to this situation is to have two separate ZPARMs and to stop/restart DB2 using the ZPARM appropriate for the type of processing required. This is not a viable approach for most large DP installations.

What Do You Really Need?

You need the ability to specify large buffer pools *and* to specify the prefetch quantity that will provide the best performance for each environment. The *second* requirement is the ability to dynamically alter the prefetch quantity (alone) while DB2 is operational. This will allow efficient transaction processing during the normal business day and the ability to change the environment for efficient processing of large jobs that often scan entire tables during overnight processing without having to stop and restart DB2. Ultimately, the system should be made *smarter* and dynamically vary the prefetch quantity and trigger point to optimize performance.

FIGURE 2



There can be a variety of interim solutions before the Optimizer reaches the level where it really *optimizes* the performance of sequential prefetch. Any of the previously mentioned options would be an improvement. Additionally, moving the point of prefetch activation lower by 16 or 32 pages (corresponding to the prefetch quantity) would reduce I/Os and improve overall performance.

ABOUT THE AUTHOR

Joel S. Goldstein is president of the Responsive Systems Company, an information systems consulting firm that specializes in on-line database systems. He has 26 years of data processing experience. Responsive Systems Company, 281 Highway 79, Morganville, NJ 07751, (201) 591-0911.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
255 (Interesting/Helpful), 256 (Too Technical), 257 (Too Basic)
2.) Would you like more articles on the same subject?
258 (Yes), 259 (No)

The Capacity Planning Database

By Michael Snyder

Capacity Planning (CP) challenges at Kaiser Foundation Health Plan and the concepts used in the applications development group to deal with the challenges was the subject of the article, "Capacity Planning: The Applications Development Viewpoint," in the February 1989 issue of MAINFRAME JOURNAL.

This article describes the CP database in detail along with some examples of its use. It also relates hard-earned practical experience.

Part 1 ended with a description of the fourfold CP problem posed by the Medical Records Management System (MRMS): it has multiple Application Functions (AFs) which must separately undergo capacity planning; it uses three different types of Workload Natural Business Unit (WNBU); it has six types of Disk NBU (DNBU); and the capacity plan changes over time because MRMS is installed incrementally rather than at all medical facilities at once.

The Database

Following is a discussion of the data stored in the workload database. There are two main record types: the company statistics record and the consumption record.

The company statistics record is used to store WNBU data for each medical facility in the company: there is one record for each medical service department within each facility, each WNBU type and each month in the planning cycle. For instance, it stores patient visit counts and patient admission counts for each facility for each month. For past months, it contains Forecast and Actual counts. For future months, it contains only Forecast counts; the Actuals are null. The data elements of the company statistics record are as follows.

- Medical facility identifier; Kaiser has 26 facilities.
- Medical department identifier. This is pertinent to activities (such as outpatient visits) that are recorded at the department level. It is null for activities (such as admissions) that pertain to the facility as a whole. On the average, there are 15 medical departments per facility.
- Period in the planning cycle. Month and year to which the record pertains. Kaiser has detailed data back to January 1985.
- WNBU type identifier. A code expressing what type of WNBU it is, such as "patient admission" or "patient visit."
- Forecast WNBU count.
- Actual WNBU count.

The consumption record is the other main record of the database. Whereas the company statistics records deal with every medical facility regardless of what AFs (if any) are installed at any given one, the consumption records deal with specific AFs which are actually installed (or firmly and specifically scheduled to be installed) at medical facilities. To further explain this distinction, consider patient admissions: the company statistics records contain admission counts for every medical facility, regardless of which facilities are using the Admission, Discharge and Transfer (ADT) system; the consumption records deal only with facilities which are *using* ADT or scheduled to be using it. The company statistics express the company activity; the consumption records express that part of the activity that imposes work on the computer system.

Being conceptually a subset of the company statistics data, the consumption data is propagated from the company statistics data. For instance, once a medical facility is placed on the ADT installation schedule, consumption records are generated from the company statistics "admissions" records pertaining to that facility from the scheduled installation date on into the future.

Capacity Planning Database

The consumption record occurs once for every AF, facility, department (if applicable), period in the planning cycle and WNBU type pertinent to the application's capacity plan. The consumption record contains the following information.

- Application Function identifier.
- Medical facility identifier.
- Department identifier. This is pertinent to activities that are recorded at the department level. It is null for activities that pertain to the facility as a whole.
- Period in the planning cycle. Month and year on which resource consumption commenced or is scheduled to commence.
- WNBU type identifier: a code expressing what type of WNBU it is, such as "patient admission" or "patient visit."
- Forecast NBU count.
- Actual NBU count.
- The network name of the CICS region sustaining the workload.

Underlying Simplifications

Batch workload is not recorded in the database. Workload imposed by certain CICS transaction IDs such as menu selection transactions is not necessarily related to any WNBU.

Due to the inherent data volume and complexity of the CP process, it is necessary to account for a minimum variety of NBUs. This introduces a certain amount of coarseness into the process.

Disk NBU accounting does not provide for the occasional necessity of leaving a major fraction of a disk volume empty in order to improve DASD response time.

To tell the truth, Kaiser has not implemented DNBUs in the CP database, because disk is easier to acquire and install than a mainframe and my CP colleagues can extrapolate from WBNU statistics. However, a pilot DNBU study was done successfully and would have been used were it necessary. A truly efficient implementation of DNBU planning will require tokenization of dataset names to AFs and a program that processes disk catalog entries, tokenizes the consumption and uses the results to update Actual DNBU data in the consumption records of the CP database.

Using The Database

Updating the consumption records to reflect changing implementation schedules and updating the Actual WNBU values in the company statistics and consumption records is where most of the effort goes in maintaining the CP database. This is a monthly job; our planning horizon is roughly two years.

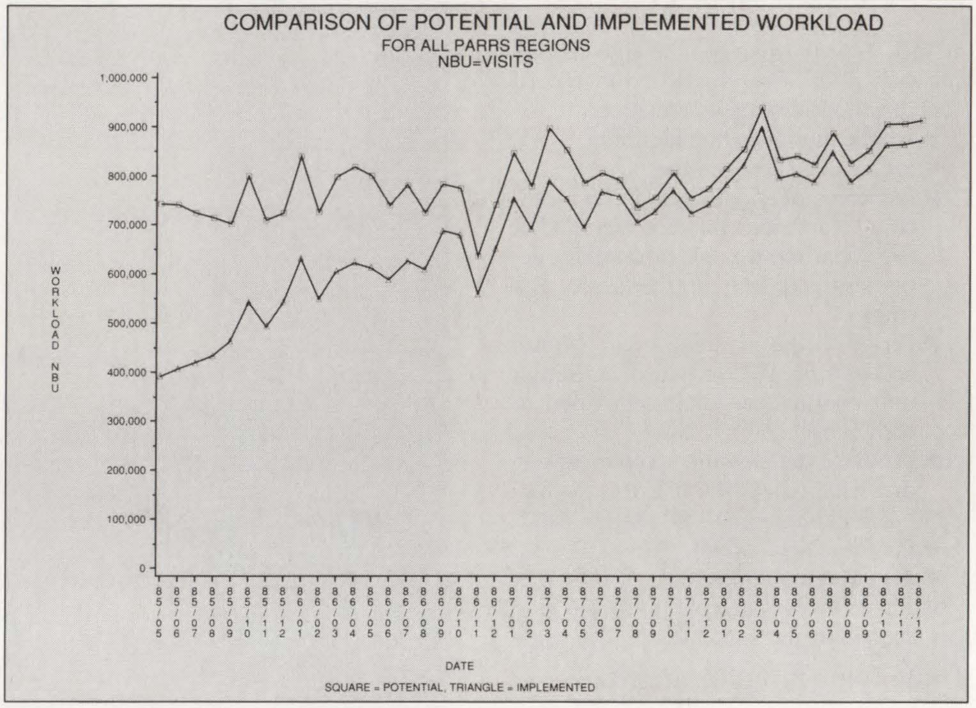
I use SAS/Full Screen Product (SAS Institute Inc., Cary, NC) and batch SAS programs to create and delete consumption records as the implementation of our major applications is scheduled, occasionally rescheduled and accomplished.

To update the various types of Actual WNBUs in the company statistics records, I use SAS/Full Screen Product and SAS files supplied by the corporate statistics office.

To propagate WNBU counts from the company statistics records to the consumption records, I use a SAS job that generates updates to the consumption records based on the data in the company statistics records.

To translate the NBU data into MIPS and megs, my CP colleagues mainly use a printed report that shows historic Actual NBU values as of a recent "CP baseline" month and future Forecast NBU values expressed as a ratio against the baseline. Using this data, they are able not only to budget the horsepower needed for application-owning CICS regions, but

F I G U R E 1



also that needed for the transaction routing resource consumption of CICS regions that are both application-owning and terminal-owning on behalf of other application regions.

Via various transformations (and a metaphysic leap of faith or two), they also made the predictions necessary to define the times at which the Patient Appointment, Registration, and Reporting System (PARRS) was successively subdivided from one to its present four CICS regions, for instance. For the future, they use the data to predict the necessity of defining new CICS regions for major new applications.

To present analogous data to management, I use SAS/Graph and a pen plotter. The accompanying plots are an example of this data for PARRS.

Figure 1 shows the company-wide patient visit workload (with the square data points) versus the visits at the facilities using PARRS (with the triangle data points). The plot shows that by the end of 1988, PARRS will be handling almost its entire potential workload; there are some medical departments yet to be implemented in 1988. Figure 2 shows the same information in terms of the percentage of the potential workload actually being handled by PARRS.

Practical Experience

Having done this work for three years, I would like to pass along a few thoughts that you may find of value in preventing capacity planning from being a high-stress job.

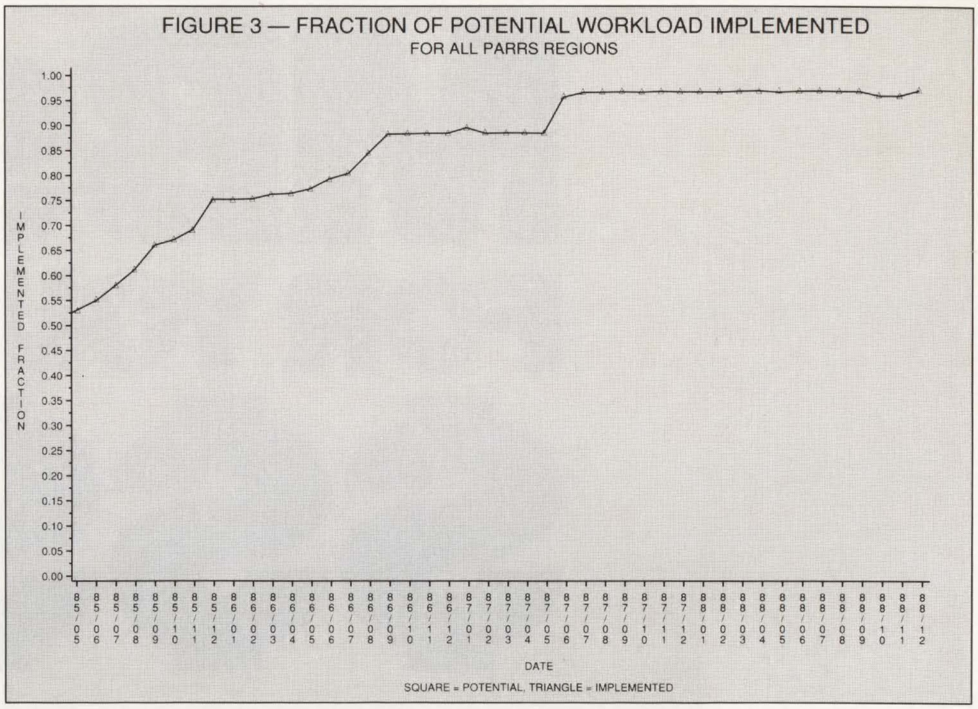
Resist the temptation to ask your business enterprise to institute any operational changes that would make the capacity planning job easier, because you will probably either be laughed at, reprimanded or stomped. It is a tough job, but it is all yours.

To avoid getting bogged down in NBU accounting, do not attempt discretely to deal with any application that will ultimately represent less than 10 percent of the total demand on the computer. For those myriad "smaller" applications, my CP colleagues simply aggregate them, look at how much power they consume today and forecast them with minor transformations into the future.

Sooner or later, one of your mathematically oriented CP colleagues will run a statistical regression that reliably tracks historic peak CICS consumption across time. Pay no attention to this whatever unless you have no new major applications coming on-line within your planning horizon.

Remember that no matter how quantitative this discipline might appear, it is fully as much art as it is science. Remind your CP colleagues of this fact whenever necessary. You might note in your own organization the fact that the applications development group, dealing as they do with end users, has a much greater tolerance for ambiguity than do your associates in the systems programming group.

F I G U R E 2



These folks, who tend to intolerance of ambiguity, have to translate all this somewhat fuzzy planning data into a tangible increment of computer horsepower and do not like to think fuzzily. All you can do is try to maintain mutual respect and realize that we are all in this together.

Once one or two years' worth of Actuals have been accumulated in your database, you can bravely compute a trend and forecast a future year by applying an observed growth rate factor to the most recent year's data, record-by-record. However, remember that if your AF is quite granular (for instance, installed department-by-department rather than in the entire medical facility all at once), you are especially subject to projecting random anomalies as trends. Such forecasts might look great, but they are implicitly dependent on the wisdom and company insight of the designer of the forecast.

Once a granular application is completely installed at a given medical facility, you could condense all of that facility's departments into a single granule for future forecasting of its consumption. So far, however, Kaiser chooses not to summarize its data so drastically. The main reason for this is that the detailed data of existing medical facility departments is useful in forecasting the NBUs of new facilities.

Acknowledge the fact that choosing truly predictive NBUs is an iterative process. After selecting one, forecasting its values over time and letting some time go by, you must find out if actual CICS workload changed by the same relative amount as did the NBU. If there is significant discrepancy, the NBU you have chosen has proven not to be a good predictor (it is in fact an Unnatural Business Unit) and you must try something else.

To find out how much work your major applications are doing, instrument them to give feedback on activities that characterize their workload. These activities need not be the same as the NBUs used in the capacity plan but will ideally relate to the NBUs *in a linear fashion*. For instance, the PARRS database allows Kaiser to readily derive appointment counts which are compared to the outpatient visit WNBU counts by which the PARRS workload is forecast. If no other feature of an application provides adequate workload feedback, consider having it create a CICS journal specifically for this purpose. (Unfortunately, this causes the measurement process to alter the thing being measured and not in the direction preferred.)

Periodically assess the variance between Forecast and Actual NBUs. They can be appallingly inexplicable but a reasonable objective in a situation like Kaiser's is this: over a period of a few months, the total NBU variance (Forecast-to-Actual pluses netted against minuses) of a given AF in a given medical facility should be small. In a multi-processor machine like the 3090, assuming it is not overall busy at peak, one way to fix a CICS region that approaches saturation of one processor is to split out the medical facilities that it serves into two CICS regions if the applications are susceptible to this solution. For this to work accurately, then, the NBU variance of each medical facility should be small so that the net effect of the split can be reasonably forecast for each of the split-out CICS regions.

A WNBU is sometimes nowhere as homogeneous a unit as you wish it were: different users can cause the computer to do startlingly differing amounts of work. Kaiser's worst-case events differ from the average by a factor of 100 (yes, one hundred) in I/Os and CPU and wait times and endemic variations of around 15 percent exist among various medical facilities for the same PARRS WNBU. Such surprises happen, for instance, when one group of users do appointment database searches under a typically constrained range of search arguments while another group will specify a broader range of eligible appointments as a normal matter of course; thereby consuming a different amount of computer horsepower for a given WNBU.

You can account for this type of WNBU "weight variance," if it is consistent, by applying a weighting factor unique to each deviant medical facility when propagating company statistics Forecast and Actual WNBUs to the consumption records — this way, the weight of a WNBU can be tailored to the practices of the user. It does introduce imponderable variables and makes the database difficult to audit for accuracy; I have never had the nerve to try it.

Another non-homogeneity problem is that with modifications to the application and its database design, the amount of resource consumed per NBU will change over time. This will make monthly comparisons of the results misleading. Nobody I know of has the persistence to try normalizing the effects of such changes over time.

In the case of a CP database design where Forecast and Actual figures are selectively copied out of a company statistics file into a consumption file, you must write a program to audit these data replications for accuracy. If you replicate and fail to audit, you will be hopelessly out of control by the time you have run less than a dozen "mass update" adjustments of your data.

When new medical facilities are built, it is necessary to predict diversion of workload from existing nearby facilities and the rate of "acceleration" with which workload will build up at the new facility. This is accomplished by wild guessing followed by upgrading of the Forecast as the Actuals become manifest.

The More General Case

There is nothing about CP with NBUs that restricts it to the health care industry.

The workload on an inventory management system in retail sales, for instance, will likely vary according to customer buying patterns and growth of the company. For instance, if there is seasonality in sales volume and/or new stores are being established, CP would be important for the inventory management application.

The WNBU for such a business would likely be either dollar sales volume or number of sales transactions. Whichever of these is regularly forecast by management for planning purposes would be the one to choose for CP.

One factor that needs to be considered for this type of application is the fact that the inventory needs to be in stock before it can be sold; if there are seasonal sales surges, the inventory management workload might increase in the month before the sales go up and taper off in the month before sales go back down. In this case, the capacity planner needs to consider time-shifting of the inventory management workload's effective date as compared to the sales WNBU effective date. If management had an inventory activity forecasting function, that might be a better source of WNBU data than would the sales figures.

Whatever the industry, if the EDP workload shows long-term change and if there are quantifiable and forecastable business events that can be related to the computer's workload, the methods described in this article can be brought to bear against the CP problem.

Conclusion

CP has proven its value at Kaiser and the intent is to keep at it until the price

See Capacity Planning *page 67*

Solid State Disk Is Making DASD Tuning A Thing Of The Past

By Dennis Corburn and Brian Fitzgerald

The attitude toward the computer during the period when large scale computers were starting to be used in commercial industry was similar to the attitude toward the automobile during its early years. Originally it was used by the technologists and scientists of the time and was considered a fickle, unruly machine with limited general application. However, over time both the automobile and the commercial computer evolved to become major forces in the growth of industry and commerce.

With the computer, that growth has been especially dramatic over the last 25 years. With IBM's System 360 Series, computers went from scientifically oriented number crunchers doing sophisticated calculations (such as plotting planetary orbits and missile trajectories) to general purpose business systems doing commercial work on huge amounts of data.

As computer systems became more integral to the day-to-day operations of enterprises, their functions went from batch jobs with few users to interactive systems supporting many users and a great variety of applications.

The evolution of computer technology has been dramatic during this period. The speed at which the computer processes information has increased more than 30 times in the last two decades. During this same period storage capacities have gone up more than 60 times. An area of less dramatic gain has been the ability of the computer to move its data from disk storage to internal storage for processing. To put it in perspective, in the same period that saw the CPU performance increase by 30 times, the speed to get the information to the processor has only increased a mere four times. To torture the car analogy a bit further, it is as if today's 18-wheeler would be forced to travel on unpaved country roads. You would have a machine capable of storing a great deal of material (data) and of processing that material at high speeds. However, its ability to get that material from place-to-place would be limited and the overall effectiveness of the system would be far less than its potential.

Such is the case with computers today. To help reduce the transportation bottleneck between the computers's tremendous ability to process data and its storage facilities, systems managers have developed a variety of storage tuning techniques. These techniques are basically designed to tune or balance the slow speed storage devices in relation to the high speed processor and limit the build-up of storage and processor requests.

One device that "throttles" these requests is a disk controller that contains a high speed storage section. This section is called "cache" and can access data many times faster than rotating disk. However, the amount of this cache storage area is limited and only data that is in the cache area will enjoy the benefit of the high speed access. Part of the control of the cache involves special caching schemes designed to anticipate which data would be required next and hold that data in the high speed electronic storage, waiting for the processor to request it.

Cached controllers illustrate the dichotomy that has existed between electronic and rotating storage devices since their introduction in the seventies. Caching attempts to combine the benefits of rotating disk (high capacity, low cost/MB) with the benefits of electronic or solid state storage (high performance) to overcome the drawbacks of each storage type; rotating disk's slow performance and electronic storage's high cost and low capacity. The problem with a cached controller is that performance depends on the correct data being held in the cache storage when the processor demands it. Caching works well under certain conditions (when the ratio of read to write requests is about 70 percent) but rather poorly in most other situations in which it is much more difficult to anticipate what data would be needed next. The relatively small sizes of the cache storage (ranging from a few thousand bytes to under 200MB compared to about 1,000MB for a typical rotating disk) limits the ability of the system to store enough data to anticipate correctly what the CPU would need.

But while systems analysts and vendors were trying to find ways to bridge the gap between slow disk and fast CPUs, another trend was occurring that could eliminate the gap altogether. While the cost and capacity of magnetic storage were improving rapidly during the seventies and early eighties, those of electronic storage were moving even faster. Since 1975, the cost for a megabyte of electronic storage has fallen 200 fold while chip capacities have risen nearly three thousand times over what they were even thirteen years ago.

By the early 1980s several companies had taken advantage of these improving price/capacity ratios in electronic storage to begin marketing Solid State Disk (SSD) products. The principle behind SSD was simple. By creating an electronic storage device with all the capacity of a rotating disk and that emulated the functions of a rotating disk, you finally eliminated the performance gulf between mechanical and electronic storage. And while SSDs were still more expensive on a cost per megabyte basis, by using them intelligently they were tremendously effective.

Since SSDs are still more expensive than rotating disk, there is no wholesale replacement of rotating disk devices with SSD devices. However, these devices are finding use as performance tools which sometimes eliminate much of the need for expensive, time consuming storage tuning efforts. For sophisticated data processing shops, SSD represents another tool for the performance specialists while in shops with less resources its sheer power can serve as a performance "cure-all."

Keep in mind that storage tuning has one main goal, to keep requests for data from backing up at the slow disk devices. When queues start forming at these devices, the CPU spends more time waiting and response time suffers. This goal is accomplished in several ways, each of which has its own drawbacks.

Before tuning the system, performance analysts need to know a great deal about how the various resources of the system are being used. These analysts can use one or more of several software applications which help them to do this. These tools look at how main storage, disk and channels are utilized. By studying this information over time, the analysts can determine where the bottlenecks are in the system and take steps to minimize them.

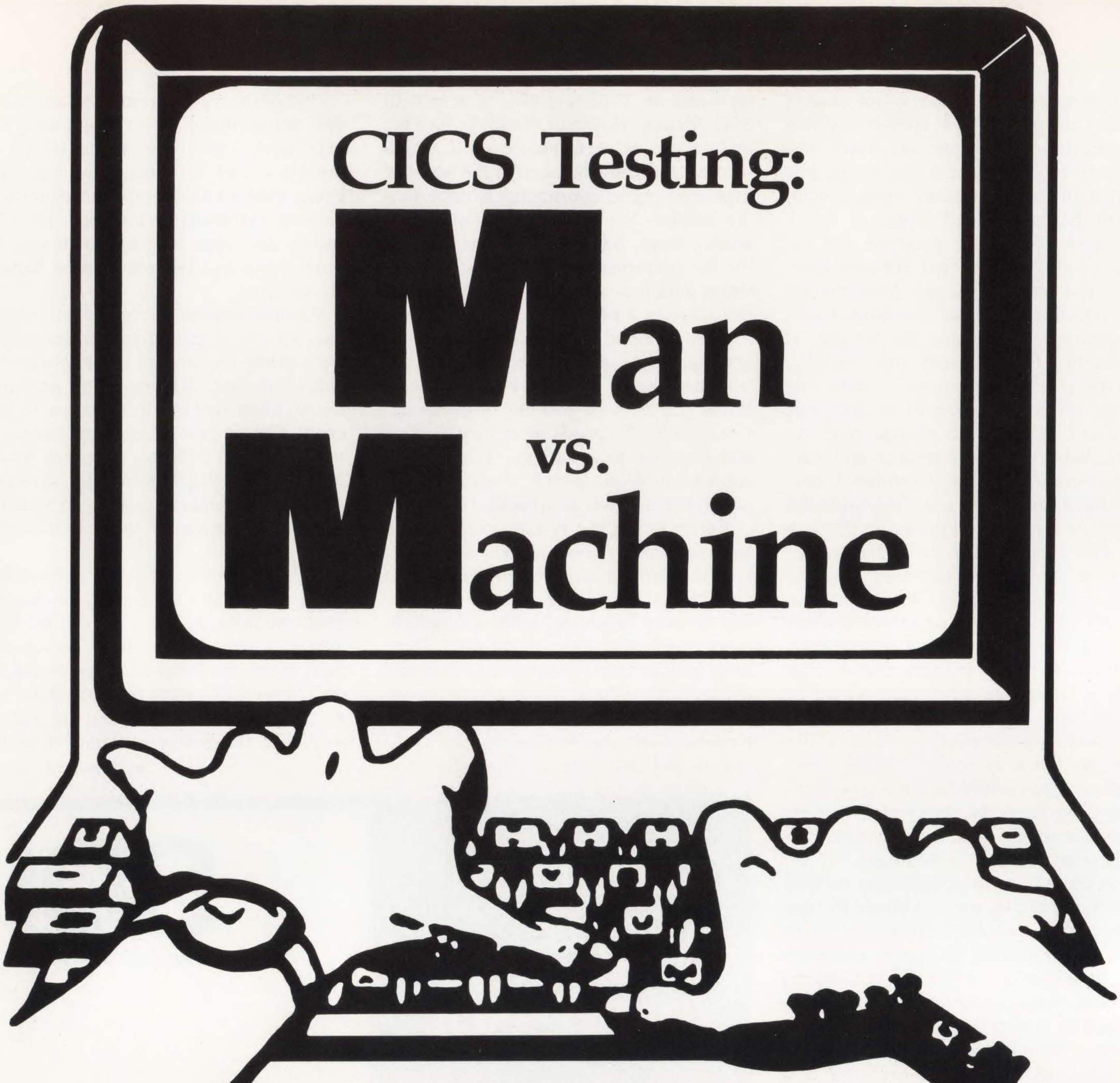
One of the most common techniques of disk storage tuning involves spreading data over many drives. The benefit of this approach is that the chance of one drive being asked to handle several requests for data simultaneously is lessened. The CPU spends less time waiting for drives, because more I/O requests can be handled at one time.

Another approach is to use only part of the capacity of each drive on the system. By limiting the amount of the platter surface being used, the distance the head must travel when seeking information is lessened. This improves the performance of the drive. But with this approach the actual cost per megabyte of disk storage is significantly higher because so much of the drive is actually unused or wasted space.

A third common DASD management tool is to increase the number of channels available between the CPU and the disk. The goal here is to maximize the chances that, when the drive is ready to transfer data to the CPU, there is an available path to transfer the data. When no path is available, the drive must wait for another

See Solid State *page 98*

CICS Testing: Man vs. Machine



By Thomas Miller

How often has inadequate testing and guess work caused havoc when changes are made to your CICS system or applications. Far too frequently we make changes, turn them over to production and pray they work when put to the real test. Wouldn't it be nice if you could know for sure what effect these changes will really have *before* they are put into production?

I can simply place most DP shops into two categories; those that try to do proper testing and those that make excuses for not doing it. In either case, adequate testing is rarely done.

The installation that tries to do testing usually falls short due to human reasons. These shops usually have a formal Quality Assurance (QA) department or procedures that are intended to guarantee quality. The quality process is usually left to the naked eye. Only errors that are visually noticed are caught. In addition, the time and expense of doing a true regression test (testing all aspects of an application or system for expected results not just those that are known to have changed) usually precludes it from being done at all. These installations understand the need for an *automated* way of testing since it simplifies their job while providing superior results.

The installations that currently do not have formal QA procedures are sometimes stubborn to have any new testing procedures, automated or not, since they seem to require additional effort on their part. This rationalization is inane. How often has their company lost money while backing out a production change during working hours?

The need for proper testing is obvious. The average installation today spends 70 percent of its time doing maintenance. Without effective quality testing, an installation is in jeopardy every time a change is made to an application or system.

The need for automated testing is just as obvious. We cannot expect humans to visually compare the abundance of data required to perform a proper test or to manually repeat every test case whenever a change is made. Software tools are available to simplify and guarantee the testing process.

Look at several different examples of manual CICS testing compared with automated testing using a QA software tool.

Stress Testing

Stress testing is used to determine how a system behaves under high loads and to determine its point of overload.

Manual stress testing is often attempted usually with little success. This can be a reasonable testing method for small or batch systems. But it is preposterous for any real-time, large-scale CICS application or system. Typically, a manual stress test requires many people to enter transactions as fast as they possibly can. Needless to say, this is labor intensive, costly and has minimal results.

With an automated testing tool you could collect large volumes of data (terminal inputs and outputs) and re-execute this data in a controlled environment. The rate at which transactions are executed can be controlled allowing you to create a realistic stress situation.

If a large volume of data is not available, you may wish to replicate smaller amounts of test or production data to create the desired effect.

Concurrency Testing

Concurrency testing is used to ensure correct results when multiple identical or similar transactions are processed simultaneously. This usually occurs when multiple terminal users execute the same transaction at the same time.

These concurrency problems are quite often the cause of many intermittent CICS production errors. They are difficult to diagnose and correct since they do not occur on a consistent basis. Quite possibly a programmer used a non-unique piece of storage (like the CWA) and did not consider CICS's ability to multi-thread programs. The first transaction uses the storage, CICS goes through dispatching and the second transaction overlays what the first did. When the first transaction regains control, the storage information it was expecting is no longer valid.

The rate at which concurrency failures occur typically corresponds to the volume of the transaction. If only one person uses the transaction, a concurrency problem will never occur. But if 50 people are using it, the chance of errors greatly increases.

The manual method of doing a concurrency test is seen frequently in DP shops. You notice it because it looks ridiculous. Usually two people prepare to enter the same transaction from different terminals (typically across the room from one another). Then they go "1 . . . 2 . . . 3 . . . ENTER." Since today's computers work in nanoseconds, the chance of these transactions really executing the same instruction paths concurrently are about as good as getting hit by lightning. There is a better chance of the first transaction completing before the second even enters CICS. The result is not knowing if your test passed or if a concurrency situation did not arise.

With an automated testing tool, you could replicate a single terminal session into a multi-terminal session and then have these transactions forced into CICS at truly the same time. A well-designed automated tool employs *virtual terminals* (terminals defined to the TCT but with no physical terminal attached) to avoid tying up real terminals.

Introducing CICS RADAR

Policing CICS region outages is a time-consuming, costly process for you, the CICS systems programmer. But now with a new product from Compuware, experienced leaders in system software solutions, you have a new weapon to add to your arsenal: CICS RADAR. With RADAR you can reduce the time it takes to debug system dumps by up to 90%!

CICS RADAR automatically pinpoints the exact location of CICS region failures, regardless of the number of regions, task activity, or stress on the system. Then RADAR produces a report that tells you what happened. You can review the report online, or print it for future reference. RADAR even provides fully menu-driven, interactive access to a compressed and formatted system dump at the touch of a key.

CICS RADAR also helps to prevent future outages with its exclusive "Snapshot" feature. This allows you to perform a complete analysis of any CICS region, before it fails, helping you to guard against unforeseen problems. And CICS RADAR is compatible with the Compuware Command Center.™

If you need speed *and* control in your fight against CICS region failures, call Compuware today about a free 30-day CICS RADAR trial. Compuware Corporation, 31440 Northwestern Highway, Farmington Hills, MI 48018. (800) 521-9353. In Michigan, call (313) 737-7300.

COMPUWARE®

Because experience is everything.

CIRCLE #67 on Reader Service Card ▲

Migration Testing

How do you test when migrating from one release of CICS to another or from one operating system to another?

The manual method usually requires a systems programmer to come in on the weekend, bring up production with the new release and then arbitrarily enter transactions to see if they work. When production is heavily used the following Monday morning, errors may arise. The systems programmer could not possibly have tested for all possibilities or recreate the full environment.

A software testing tool could capture the entire previous day's production load. Then the systems programmer could re-execute this load after he installs the new release. Reports can be produced automatically outlining any differences found between the two versions.

Unit Testing

Unit testing is the most common type of testing today. This is the testing of an individual unit of work (typically a change to a program) to make sure it performs as it was designed.

A manual unit test is quite simple. The programmer calls up the screen or screens that were changed and visually looks to make sure the change is correct. He usually checks the entire screen to make sure his change did not alter other fields. Then he mentally makes a decision, "Could the simple change I just made possibly affect anything else in the system?" He weighs the options, "If I were to test my entire system, it would take all day" vs. "Na, my change was simple, it could not possibly affect anything else." The latter almost always wins. Then when the program migrates into production, an error is found in a different transaction, perhaps because these two transactions share the same common file.

With an automated testing tool, you can not only have it unit test your simple change, but also have it automatically check all other possibilities. It will report back to you on any differences found between the screens prior to the change with the screens after the change. This way you are guaranteed a true regression test.

I think it is apparent that an automated approach to CICS testing is far superior to any manual procedures. Quality should always be foremost in the minds of your DP staff. And to provide this quality, a powerful QA software testing tool is essential.

ABOUT THE AUTHOR

Thomas Miller is the product manager for On-Line Software's VERIFY, a leading CICS quality assurance system tool. He is a member of the Quality Assurance Institute and has had extensive experience in areas ranging from system performance and tuning to application design and coding. On-Line Software International, Inc., 2 Executive Dr., Ft. Lee, NJ 07024, (201) 585-6656.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
270 (Interesting/Helpful), 271 (Too Technical), 272 (Too Basic)

2.) Would you like more articles on the same subject?
273 (Yes), 274 (No)

Capacity Planning Database

Capacity Planning *from page 61*

and availability of hardware does not matter any more. The CP group is far from making its retirement plans.

ABOUT THE AUTHOR

Michael Snyder is senior technician in the Application Development Group at Kaiser Foundation Health Plan, Inc. His background includes 20 years of application development concentrating on analysis, design and coding.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
260 (Interesting/Helpful), 261 (Too Technical), 262 (Too Basic)

2.) Would you like more articles on the same subject?
263 (Yes), 264 (No)

CICS Quality Assurance/System Testing Tools Sampler

Product	Vendor	Vendor Comments	Environment	More Information
CICS PLAYBACK	COMPUWARE 31440 Northwestern Hwy. Farmington Hills, MI 48018 (800) 521-9353 In Michigan (313) 737-7300	CICS PLAYBACK is a comprehensive, practical quality assurance system that allows you to test your systems before your users do. It provides powerful capabilities to completely test CICS software during normal working hours without using the network. This ability without the physical network eliminates high network and personnel costs. CICS PLAYBACK provides the flexibility to repeatedly test CICS systems, without entering data for each test, until quality assurance goals are achieved. It can save 70% of the staff time required for testing.	MVS, VSE	CIRCLE 226
CICS/REPLAY	INTERACTIVE SOLUTIONS, INC. 53 W. Fort Lee Road Bogata, NJ 07603 (201) 488-3708	CICS/REPLAY is a highly advanced automatic CICS regression testing and quality assurance package that enables developers and technical support personnel to capture production transactions or easily create test data and see the effect of changes in production while in test. It lets you review, perfect and execute CICS applications and releases on-line automatically in just one session. This ensures a clean cut over into production by a single person and significantly increases total development and technical support productivity.	MVS, VSE	CIRCLE 227
VERIFY	ON-LINE SOFTWARE INTERNATIONAL Fort Lee Executive Park Two Executive Drive Fort Lee, NJ 07024 (800) 526-0272 In New Jersey (201) 592-0009	VERIFY is an on-line, menu-driven system that automates CICS application and system testing tasks, so you can eliminate lengthy and error-prone manual techniques. With VERIFY you will know immediately whether your program will produce unexpected results. It lets you see and/or resolve any differences that arise during the testing of new and modified applications. Its interactive capabilities let you easily handle any run-time testing discrepancies.	MVS, VSE	CIRCLE 228

Understanding The Subsystem Interface Component of MVS

By Bruce Bordonaro

A subsystem, loosely defined, is a set of programs and/or routines defined to the Subsystem Interface (SSI) component of MVS to provide services or functions not directly provided by basic MVS facilities. The SSI is a branch entry interface available for any service facility to use. It is most typically used by Job Entry Subsystems (JES').

The SSI came about with the advent of the MVS operating system to provide a common interface for communication between MVS and JES. Prior to MVS, hooks were placed at various points in operating system code by HASP and ASP, the forerunners of JES2 and JES3, respectively. Both, for example, had hooks placed in the Input/Output Supervisor (IOS) to intercept spool I/O and redirect it to the appropriate spooling space.

The SSI was developed to eliminate the need for the hooks described above so that MVS could make common calls through a standard interface to subsystems for notification of events, to provide additional service and to monitor work in the system. Through this method various functions such as SYSIN/SYSOUT processing, job selection/termination and the use of internal readers can be provided via the same stable system exit points. Even though the JES' were the original reason for the design of the SSI, it was also designed flexibly enough to enable general use by subsystems yet to be developed at the time such as IBM's IMS, DB2 and BDT and other non-IBM products such as TMS, LOOK and LIBRARIAN.

One of the more popular uses of the SSI is in automated operations.

Subsystem Specification

All subsystems must be identified to MVS by a name of four characters or less. The first character must be alphabetic or national and the remainder either alphanumeric or national. Prior to MVS/SP 2.2.0, there were three ways in which a subsystem could be defined:

- The PRISUB and SUBSYS keywords of the SYSGEN SCHEDULR macro
- A zap or link edit of the subsystem name table (IEFJSSNT)
- The IEFSSNxx PARMLIB member.

As of MVS/SP 2.2.0, only the last method listed above is supported to add subsystem entries. This method was first introduced in MVS/SP 1.2.

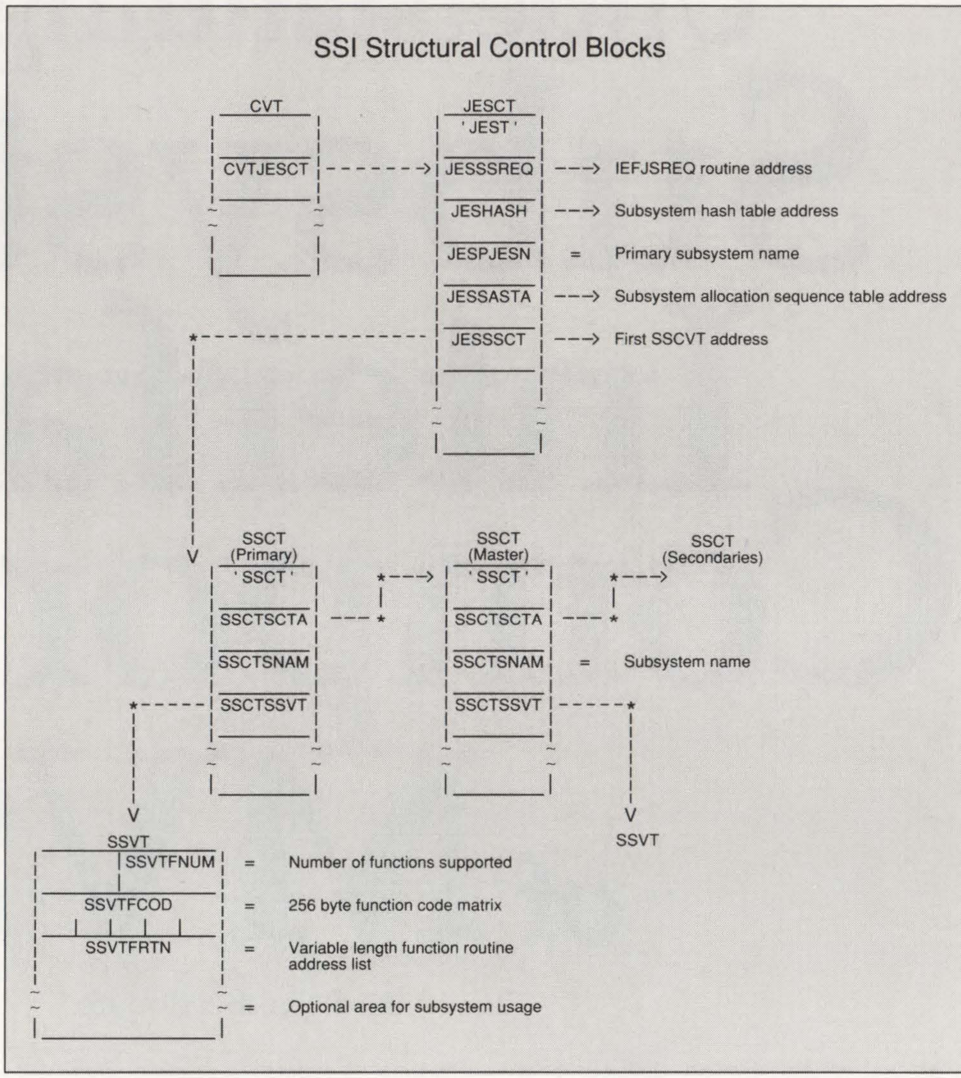
In the PARMLIB member or in IEFJSSNT, there is the option to specify the name of a subsystem initialization program that will be invoked during master scheduler initialization; this routine must reside in the linklist concatenation. In the PARMLIB member only, specify an optional parameter string to be passed to the subsystem initialization program thus providing the flexibility of being able to change some subsystem options at startup time. The initialization routine receives control in supervisor state with register one pointing to a two-word parameter list containing the address of the Subsystem Communications Vector Table (SSCVT) followed by the address of the initialization parameter list that is mapped by the IEFJSIPL macro. The subsystem initialization program method is usually used when the subsystem will not have an address space resident in the system for the life of the current IPL. If this method is not used, then a job must be started to initialize the subsystem. That job must bear the name of the subsystem and the JCL to start it must reside in the procedure library, usually SYS1.PROCLIB, pointed to by the IEFPDSI DD statement in the master scheduler's JCL (load module MSTJCL00 in SYS1.LINKLIB). Any datasets referenced by the procedure must either be cataloged in the master catalog or pointed to by specific unit and volume serial number.

The master subsystem is defined automatically by the SYSGEN SCHEDULR macro and it is called MSTR. It does not run in its own address space, rather its support is provided by routines in the Link Pack Area (LPA). Its job is to initialize the master scheduler as well as other subsystems that will not run under the primary job entry subsystems. It does this by reading the JCL in the MSTJCL00 member of SYS1.LINKLIB and passing it through the converter/interpreter using a pseudo access method interface instead of the normal VSAM access method.

The pseudo access method manipulates data in real storage rather than external storage as normal VSAM does. Since the pseudo access method uses the regular RPL/ACB interface, the converter/interpreter can use the pseudo access method just like it was VSAM. The master subsystem is also responsible for "broadcasting" certain SSI requests to all other active subsystems (which will be discussed later).

The primary subsystem is defined by the PRISUB keyword of the SCHEDULR macro or with MVS/SP 2.2.0 in PARMLIB via the PRIMARY keyword as the fourth positional parameter of the subsystem specification. In addition, you may indicate that you do not wish the primary subsystem to be started automatically as the fifth positional parameter, such as:

F I G U R E 1



JES2,,,PRIMARY,NOSTART

Only one primary subsystem can be defined to MVS and it must be a job entry subsystem. You may also specify the name of an initialization routine for the primary subsystem. The primary will control the processing of START, MOUNT and LOGON requests as well as the handling of SYSLOG. Note that START requests also include initiators under which batch jobs will run. Prior to MVS/SP 2.2.0, the primary was able to be started by a command embedded in the master scheduler JCL (MSTJCL00). As of MVS/SP 2.2.0, if an IEFSSNxx entry contains the PRIMARY parameter, the subsystem will be started automatically unless the NOSTART parameter is also coded; then the subsystem must be started by operator command.

Secondary subsystems are entirely optional but are becoming increasingly more commonplace. They may or may not run under the primary job entry subsystem, depending on whether or not they require SYSIN/SYSOUT services. Note that JES2 supports running itself as a secondary subsystem while JES3 does not. In this case, started tasks and batch jobs may be directed to the secondary JES2 by means of the SUB=keyword of the MVS START command. Normally, TSO logons are only directed to the primary JES, but there are some MVS user modifications on the SHARE and/or Connecticut Bank and Trust (East Hartford, CT) MVS modifications tapes which allow TSO logons to be handled by a secondary JES.

SSI Structural Control Blocks

Certain control blocks required for the SSI are created during the IPL process. The JES Control Table (JESCT) is created residing in key zero NUCLEUS storage and is pointed to by the CVTJESCT field of the Communications Vector Table

(CVT). It is mapped by the IEFJESCT macro. The JESCT contains some general subsystem information such as the name of the primary job entry subsystem (JESPJESN), pointers to the routines that handle SSI requests (JESSSREQ) and other information relating to device allocation and the SSI.

The SSCVTs are created for each subsystem defined by the methods mentioned above and are chained together via the SSCTSCTA field of the SSCVT. The SSCVTs are created in subpool 241 key zero storage and the first SSCVT in the chain (which would be the primary JES) is pointed to by the JESSSCT field of the JESCT. The end of the chain is indicated by an SSCTSCTA value of zero. Each SSCVT is mapped by the IEFJSCVT macro.

When a subsystem is initialized, it is the subsystem's responsibility to create a Subsystem Vector Table (SSVT), usually in some common storage area. The SSVT is pointed to by the SSCTSSVT field in the SSCVT and consists of a 256-byte function code matrix indicating what function codes are supported by the subsystem, followed by a variable size table of addresses of the subsystem function routines themselves.

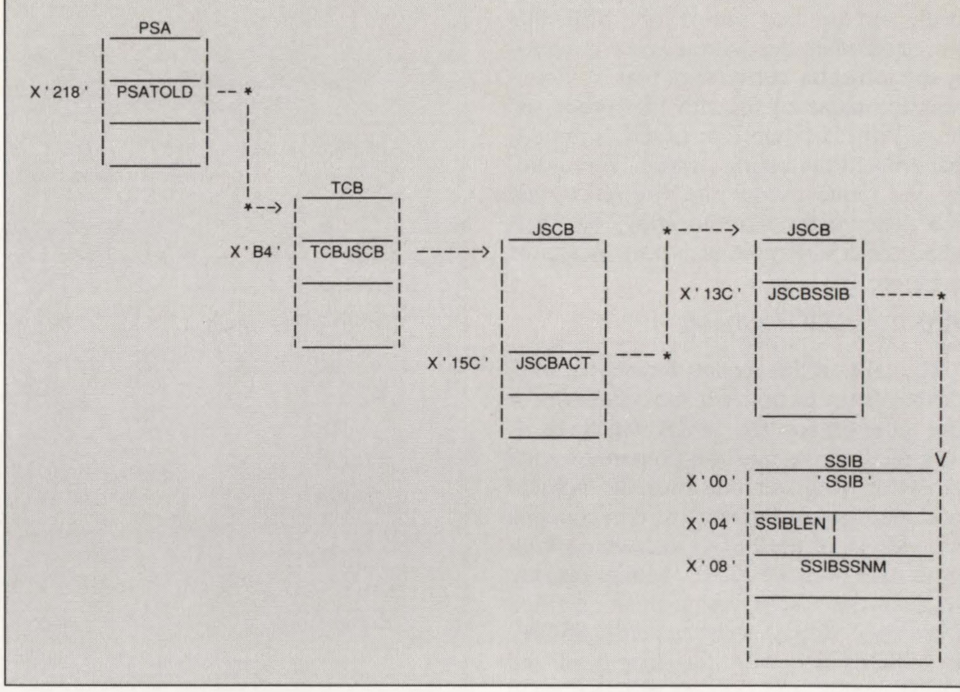
Each basic SSVT is mapped by the IEFJSSVT macro. Optionally, there may also be a variable length extension following the function routine addresses which may be used by the subsystem for other unique information. JES2 and JES3 have additional material appended to the end of their SSVTs. For JES3, it is mapped by the IATYSVT macro; for JES2 the mapping macro is SSVT. When the address of the SSVT is filled into the SSCVT, the subsystem is considered active and able to process requests.

Also as part of the IPL process, two other related control blocks are created. The subsystem hash table (HASH) is built and pointed to by the JESHASH field of the JESCT. This table is used as a quick index for finding SSCVTs but is only useful if there are a tremendous amount of subsystems. Its purpose is to eliminate the long chain searches usually associated with what IBM terms the "large systems effect."

If the subsystem is not found via the hash table, then the system will sequentially search all the SSCVTs to find a match since other SSCVTs could be created and added onto the SSCT chain after the IPL process. The Subsystem Allocation Sequence Table (SAST) is also

F I G U R E 2

Finding SSIB When SSOB Pointer Is Zero



pointed to by the JESCT, specifically the JESSASTA field, and is used to control the order in which subsystems will be given control to process subsystem allocation requests, that is, those DD statements containing the SUBSYS=keyword. The relationship of the control blocks described above is depicted in Figure 1.

SSI Request Control Blocks

The subsystem interface is invoked via the IEFSSREQ macro instruction with register one pointing to a one word parameter list containing the address of a Subsystem Option Block (SSOB). The SSOB and its extensions are mapped by the IEFJSSOB macro. The extensions are used to hold function dependent information while the header contains the SSI request function code, an SSIB pointer, an SSOB extension pointer and a field in which the SSI return code is placed.

Proper expansion of the IEFSSREQ requires that the mapping macro instructions for the CVT and JESCT (IEFJESCT) must be included in your program. The macro call generates a branch to the IEFJSREQ routine, located in LPA, that validates the control blocks passed that provide the information as to the particular subsystem and function requested. This routine runs in the same protect key and state as the caller. The SSOB defines the function requested and also points to a Subsystem Information Block (SSIB) that contains the name of the subsystem that service is being requested from along with other function dependent information. The SSIB is mapped by the IEFJSSIB macro.

IEFJSREQ ensures that the subsystem exists, is active and supports the function code requested. If so, the value of the function code byte is used as an index into the variable size table of function routine addresses in the SSVT to pick up the address of the routine and control is passed to it. If the request is invalid, an appropriate return code is sent to the caller indicating that the function code is not supported by that subsystem.

In order to accommodate SSI callers who may be in either 24-bit or 31-bit mode, IEFJSREQ checks to see if there is a difference between the caller's addressing mode and the addressing mode of the subsystem function routine and saves the mode if necessary. Then, before giving control to the appropriate SSI routine, IEFJSREQ points register 14 to the address of an addressing mode switch routine so that upon return from the function, the correct addressing mode can be restored. If there is no difference between addressing modes then the return register is loaded with the caller's normal return address.

The caller has the option of pointing to an SSIB he has created or leaving the pointer as zero. If it is zero, the SSI will locate and use the "life-of-job" SSIB that is created when the address space is started by the job entry subsystem; that is, it contains the name of the JES that processed the START, MOUNT or LOGON request that created the address space. Therefore, the SSI request would be directed to that JES. The method for locating the SSIB when the SSOB pointer is zero is shown in Figure 2.

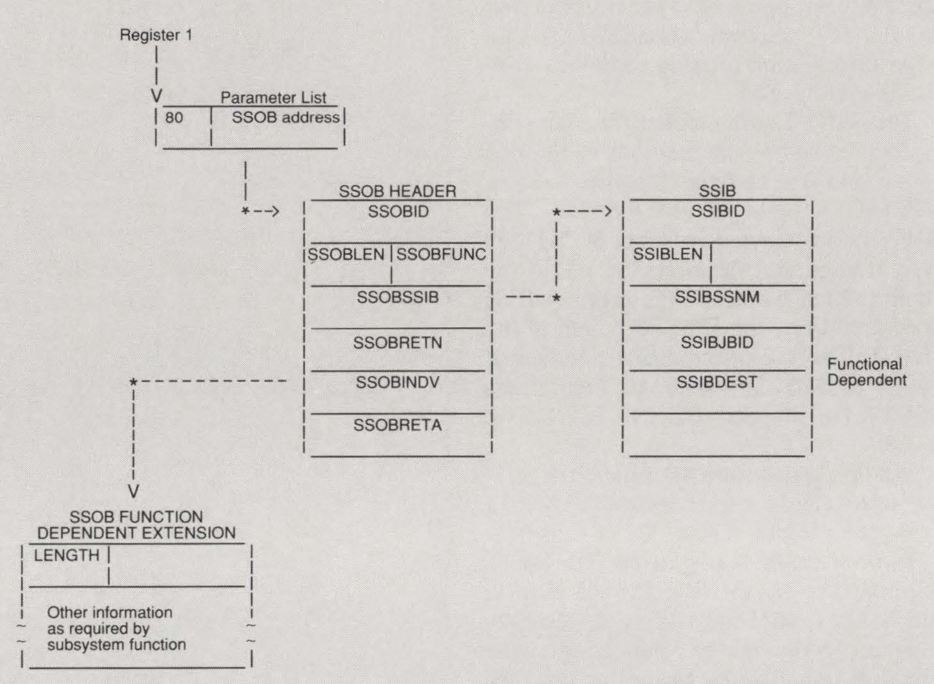
Making SSI Requests

To make an SSI request for service from a subsystem, to notify a subsystem or for any other reason the SSOB and SSIB as described above must be prepared by the requestor. Register one must be pointed to a one word parameter list that contains the address of the SSOB and whose high order byte must be X'80'. Minimally, the fields in the SSOB which must be filled in are the SSOBID with the value 'SSOB', the SSOBLEN with the length of the SSOB header and the SSOBFUNC with the function code being requested from the subsystem. Optionally the SSOBSSIB may be filled in with a pointer to the SSIB built by the requestor and the SSOBINDV with a pointer to the SSOB extension if required by the particular subsystem function. If the caller provides an SSIB, the fields which must be filled in are the SSIBID with the value 'SSIB', the SSIBLEN with the length of the SSIB and the SSIBSSNM with the name of the subsystem to which the request should be directed.

Prior to actually issuing the IEFSSREQ macro, the caller might have to change his running environment as per subsystem needs. That is, the subsystem may require a different protect key or state than those of the caller. Also, certain locks might need to be obtained. The caller must point register 13 to a standard OS save area, it must be running in TCB versus SRB mode if the "life-of-job" SSIB is to be used since it must be located through the current TCB pointer via the PSA, it must not be running disabled and must allow page faults since the SSI control blocks are usually found in pageable storage. When these conditions are satisfied, the caller can point register one to the address of the parameter list and issue the IEFSSREQ macro.

F I G U R E 3

Control Blocks For Invoking The Subsystem Interface



Upon return from the SSI processing routine (IEFJSREQ), register one will contain the SSOB address and register zero will contain the SSCVT address. Register 15 will contain the IEFJSREQ interface return code and the SSOBRETN field of the SSOB will contain a return code from the subsystem itself if the request was successful. Figure 3 illustrates the relationship of the SSI request control blocks. The possible interface return codes are:

- 0(SSRTOK) if the request went to the subsystem successfully
- 4(SSRTNSUP) if the subsystem does not support the requested function
- 8(SSRTNTUP) if the subsystem exists but is not active
- 12(SSRTNOSS) if the subsystem does not exist
- 16(SSRTDIST) if there is a disastrous error
- 20 (SSRTLERR) if there is a logical error (like an invalid SSOB/SSIB format, incorrect length and so on).

The labels in parentheses above are generated by the SSOB mapping macro IEFJSSOB. They may be used to check the interface return code returned.

Details of SSI Request Routing

When IEFJSREQ gets control, it validates the formats of the SSOB and the SSIB. If SSOBSSIB is zero, it locates the "life-of-job" SSIB by tracing through the PSA, current TCB, JSCB, active JSCB to the SSIB. Next, IEFJSREQ determines the correct SSCVT to use by dividing the name of the subsystem by the number of entries in the subsystem hash table and using the remainder as an index to the hash table. The hash table entry is either the address of the SSCVT required or the address of a synonym SSCVT that can be traced by a synonym chain (SSCTSYN) to find a match on subsystem name. The synonym search is necessary since many subsystem names may hash to the same value; all the names hashing to the same value are therefore chained together to ease the search. If this method does not yield a match, the SSCVT chain is searched sequentially starting from the last SSCVT built at IPL time whose address is stored in the hash table. This will locate any SSCVTs built dynamically after the IPL process.

The SSVT pointed to by the SSCVT is then checked to determine if the subsystem supports the function code requested. This is done by subtracting one from the function code and adding the result to the

Service level management began with OMEGAMON®. So we precision-crafted automated operations to fit perfectly with OMEGAMON. Because automation is just another part of performance management.

AF/OPERATOR™ learns about MVS, CICS, IMS, or DB2 problems from OMEGAMON. So as soon as an exception is triggered, AF/OPERATOR can issue commands to pinpoint the problem. Then speed a message directly to your best troubleshooter for fast resolution. It can even take action itself to improve service at machine speed.

Maybe the CICS claims application is way over its service level. AF/OPERATOR can issue OMEGAMON commands to determine which resource is delaying claims. Or if other jobs are causing the delay. It can then automatically track down the user to see if the conflicting job can be cancelled.

Of course, AF/OPERATOR lightens the load of your operations staff with console message management, automatic startup of time-dependent activities, reliable IPLs, and orderly shutdowns. And it simplifies your operations by becoming the single point of control for all your systems — MVS, IMS, and JES to name a few.

For over ten years, data center managers have trusted the performance of their systems to Candle. Now you can automate your data center with the same confidence. Expect AF/OPERATOR to fit your data center perfectly — now and in the future. For more information, call Terry Forbes at **(800) 843-3970**.

!Candle®

For prerecorded information 24 hours a day, call (800) 228-0631.

CIRCLE #120 on Reader Service Card ▲

address of the function code portion of the SSVT. The function code byte pointed to indicates whether or not the function is supported. If it is zero, the function is not supported and a return code is given to indicate that. If non-zero, then one is subtracted from the value stored in that byte, multiplied by four and the result is added to the address of the beginning of the function routine address portion of the SSVT. This points to the address of the function routine that will process the function code requested. At this point, IEFJSREQ will determine if the caller of this SSI request is in 31-bit mode and the function executes in 24-bit mode or vice versa. If so, register 14 is set up so the function routine will return control to the SSI mode switch routine so the correct mode can be restored. If there are no addressing mode conflicts, then register 14 is loaded with the caller's return address. Finally, register 15 is loaded with the address of the function routine that is invoked via a BALR instruction since IEFJSREQ is simply a branch entry interface.

When the subsystem function routine completes its processing, it sets register 15 to zero to indicate that the function was accepted by the subsystem. The SSOBRETN field is used to store the return code returned by the subsystem and then the routine returns to its callers or to the SSI mode switch routine if necessary.

Broadcasting SSI Requests

There are certain SSI function codes that are defined to SSI to be "broadcast" to all active subsystems. This process is handled by the SSI common request router IEFJRASP that is a load module in LPA. Only certain SSI requests can be broadcast by IBM convention. Some are notification of end of task, WTO, command processing and end of memory to name a few. A list of all the SSI function codes, what subsystem uses them and if they are defined to be broadcast can be found in the IBM System Logic Library manual that documents the subsystem interface.

The request is made like a regular subsystem request except that the subsystem name in the SSIB must designate the 'MSTR' subsystem. When the lookup is done to find the address of the master subsystem routine to give control to, the address found is that of the common request router IEFJRASP. IEFJRASP first makes a local copy of the SSOB and the SSIB. It then scans through the SSCVT chain to find each active subsystem by checking for a non-zero SSVT pointer. For each active subsystem except the master subsystem, it copies the subsystem name into the SSIB copy and issues the IEFSSREQ macro to send the request to that subsystem.

Upon return from each invocation of the IEFSSREQ macro, IEFJRASP looks at the return code returned from the subsystem. The lowest return code from the subsystem interface is stored in register 15 and the highest subsystem return code is stored in SSOBRETN. That is, if one or more subsystems handled the request, the highest return code from the subsystems that processed the request is placed in the original caller's SSOBRETN field.

Uses for the Subsystem Interface

Some of the current users of the SSI are those vendors whose products deal with automated operations. By using the SSI broadcast functions, these popular products can intercept WTOs and act upon them, reply to WTORs and simplify operator command sequences. A well known example of such automation is one that many people take for granted and that is the short form reply. When you reply to a WTOR by typing xxTEXT (where xx is the reply id), it is translated to the recognizable MVS format REPLY xx,TEXT by that handy automated operations tool called JES2.

Another use for the SSI is not really using it for subsystem communication. Many users have, over the course of time, used a CVT field available to the user (CVTUSER) as an anchor point for various homegrown or vendor products. Lately, many vendor products have started using SSCVT entries as their anchor points in the system without actually establishing themselves as subsystems. This provides a product with a unique globally addressable control block to anchor themselves and for storage of information. Some examples of this are the CA-7 (formerly UCC-7) scheduling package and the CA-1 (formerly UCC-1) tape management system. They both take advantage of two words available to the user in their SSCVTs (SSCTSUSE and SSCTSUS2) that may be used to store the addresses of tables and executable code. In both of their cases, they use an initialization routine name in their PARMLIB IEFSSNxx entry to set up the data they need and point the user words to it. In the case of CA-1, there is also a started task that can update certain information in the tables or can enable or disable tape management processing at will. This also allows such products to restart themselves by reusing previously loaded information without an intervening IPL.

By now you are thinking that life is complex enough without having to deal with the SSI. You are probably right but note that IBM does provide a semi-documented interface to set up the necessary SSI control blocks required to establish a subsystem. The subsystem service routine IEFJSBLD can be used to build SSCVTs, SSVTs, the SAST, the HASH, enable or disable SSVT function codes and build the primary subsystem's SSCVT. The routine is also used at IPL time to initialize a subsystem by building the subsystem initialization parameter list mapped by the IEFJSIPL macro and then linking to the subsystem's initialization routine. The sparse documentation for its use can be found in *MVS/XA System Initialization Logic* manual (form LY28-1200). Be warned, however, that a trip may be required to your microfiche viewer to investigate the source code for IEFJSBLD to really be able to use its services. For instance, the logic manual shows a subsystem service routine parameter list (SSPL) as input to IEFJSBLD but there is no mapping macro for it. It is also unclear that before calling IEFJSBLD, register one must be pointed to the address of a pointer to the SSPL rather than just pointing register one to the SSPL directly. A mapping macro is available for another control block pointed to by the SSPL, the SSVT table data parameter list (IEFJSBVT).

In order to get you on your way to using the subsystem interface, a piece of skeleton code that uses the MSTR subsystem's Verify Subsystem function code to determine if a subsystem exists and is active is available from MAINFRAME JOURNAL by circling number 300 on the Reader Service Card. Function code 15 is used to verify subsystem existence. If the subsystem exists, upon return from the SSI call both register 15 and the SSOBRETN field will contain zero return codes.

In the SSOB extension, the SSVSSCTP field will contain the address of the subsystem's SSVCT and the SSVSNUM field will contain the relative position of the subsystem on the SSCVT chain. If the subsystem does not exist, the SSI return code in register 15 will be zero and the SSOBRETN field will be set to four. This whole process could also be performed by

scanning the SSCVT chain yourself. However, it is my belief that if a documented interface exists, it should be used. Additionally, since the SSI common interface uses the subsystem hash table to perform subsystem name lookups, it is slightly faster than doing it yourself.

Where Do We Go Now

For gruesome detail on the structure and function of the SSI, look at the *MVS/XA System Logic Library: Master Subsystem/Subsystem Interface* manual (form LY28-1720). For a comprehensible explanation, however, I suggest chapter nine of the Science Research Associates (SRA) (Chicago, IL) reference text *A Guide To Using MVS/XA Interface Facilities* form SR21-1468. SRA was previously the educational subsidiary of IBM but is now an independent company. Other sources of information are SHARE and GUIDE presentations on the use of SSI, the MVS/XA debugging handbooks for control block formats and the Connecticut Bank and Trust MVS modifications tape that contains a copy of TSSO (a public domain automated operations tool that has existed for many years). Finally, start investigating automated operations tools since they are big users of the SSI; the vendors may be able to offer insight into the SSI since they themselves have already gone through the pain of this research.

ABOUT THE AUTHOR

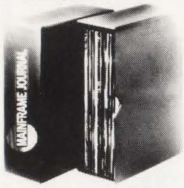
Bruce Bordonaro is a systems software manager for Pershing, a division of the brokerage firm Donaldson, Lufkin and Jenrett, New York, NY. He has more than 12 years of experience in the MVS data processing environment, 10 of which has been in systems programming.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
275 (Interesting/Helpful), 276 (Too Technical), 277 (Too Basic)

2.) Would you like more articles on the same subject?
278 (Yes), 279 (No)



JES3
Is It Worth The Conversion Costs?

By Jon E. Pearkins

There are some major advantages that JES3 has over JES2, yet it only costs four to 12 percent more. But is it worth the conversion costs if you already have JES2?

First, a more fundamental question needs to be answered. Why does IBM offer two Job Entry Subsystems (JES)? Both products were created for OS/360 on the IBM System/360 in the mid-1960s and both had similar acronyms just as they do now. However, they originally had very different names: what is now JES2 was called the Houston Automatic Spooling Priority system (HASP) and JES3 was known as the Attached Support Processor system (ASP). Both were designed to provide SPOOLing (Simultaneous Peripheral Output On-Line) so that the relatively slow speed of card readers and printers did not become a major bottleneck for system throughput.

All the punch cards for a batch job could be completely read in prior to execution and all printed output could occur after execution was complete. Since memory was expensive and the number of batch jobs executing simultaneously was severely restricted (often just one at a time), batch jobs could execute more quickly, not having to wait as each punch card was read in and each line printed.

A 1964 IBM study concluded that two different spooling systems were justified: HASP, for a single processor environment; and ASP, designed with the multiprocessor environment in mind.

System/360 users were typically divided into "commercial" and "scientific" installations. Initially, HASP was used by the commercial customer and ASP by the scientific. With the introduction of MVS in January, 1973 to exploit the virtual storage architecture of the System/370, the two spooling systems were brought closer and closer together in both functionality and name. Today, with JES2 supporting a multisystem shared spool environment and JES3 providing a superset of JES2 functionality for single systems, they both cover the same ground.

According to IBM, not only 10 percent of MVS installations use JES3 but those same installations account for 40 percent of IBM's revenue base in the MVS arena. Although small in number, JES3 shops are generally large. However, this is more a result of history and IBM marketing than common sense. As we will see, you do not have to be huge to benefit from JES3.

JES3 Approach

In an installation using more than one distinct system to share the workload (that is, loosely-coupled processors, not just multiple CPUs in one cabinet), JES3 provides a "single-system image." A single set of spool queues is maintained on the global processor, controlling the input received and jobs and output dispatched to the local processors. If the global processor fails, another processor can take over without interrupting the work being done by the local processors.

JES2, on the other hand, is capable of sharing one set of spool queues through Multi-Access Spool (MAS) but each system must do its own scheduling.

When large corporate takeovers create amalgamated data processing installations with both JES2 and JES3, the choice is invariably JES3 even when the JES2-based portion is considerably larger. JES3 is just better suited for a large multiprocessor environment.

JES3 Advantages

Some of JES3's advantages over JES2 include:

- Improved throughput in a multisystem environment by utilizing centralized scheduling
- Improved throughput in any environment because jobs are not dispatched until all required resources are available
- Workload balancing through generalized main scheduling
- The global processor operator's console provides centralized control for a multisystem environment, eliminating the need to have operators monitor each system's console
- Deadline scheduling provides an automatic means of assuring a job will complete by a specific time rather than relying on operators to adjust priorities to meet deadlines
- Dependent job control allows the scheduling of one or more jobs based on the successful completion of a prerequisite job.

If you are familiar with JES2, you may notice a lot of features of JES3 that sound more like a scheduling package than a spooling package. In fact, some JES3 installations exploit its scheduling features and find that they can live without the purchase of additional scheduling software. Coincidentally, just as I was completing this article, a local JES2 installation indicated they were planning to evaluate JES3 along with the traditional scheduling packages in their search for user scheduling of batch jobs.

Perhaps the most visible difference between JES2 and JES3 is one that positively impacts programmer productivity. Both JES2 and JES3 detect syntax errors in JCL prior to executing a job, providing a major boost in programmer productivity for installations converting from VSE and/or VM. But JES3 goes a step further by checking all datasets prior to executing a job as part of the resource scheduling process. This means that a job will not run if input datasets do not already exist unless they will be created by a preceding job step. Likewise, jobs that try to create output datasets that already exist are not allowed to execute.

JES3 dataset checking also affects production batch jobs. Non-existent and previously-created datasets are major causes of production job abends. Both application and technical support programmers know the difficulties involved in rerunning batch jobs that abend half way through. Rerunning the job from the start may result in double updating of master files. Restarting from the *step* that failed may work but usually is not all that is required. Temporary datasets created by previous steps of the job may have to be recreated for use in this and subsequent steps. Conversely, the failing step may have created datasets that need to be deleted before rerunning. So, anything that can reduce the number of batch jobs that abend in the middle of the job will improve both operations and programmer productivity. The programmer has less work to do to (re-)run the job. And since it takes less time to fix the problem, operations does not fall so far behind on its schedule.

Dependent job control and deadline scheduling also do their part for productivity by reducing operator intervention and increasing system throughput. A large series of jobs can be scheduled with JES3.

However, it goes a step further than that. All tape volumes used in a job are premounted and verified before the job begins. If a tape is unavailable (for example, off-site), the job can be cancelled or deferred and run later without having to worry about rerun problems. On the other hand, with JES2, deciding to wait for a missing tape could tie up an initiator for hours as well as any exclusively-controlled datasets (DISP = OLD) used by the job. To overcome this problem, many JES2 installations opt to have jobs automatically cancelled when tape mounts remain outstanding for more than 30 minutes. Unfortunately, this creates another problem — that of abends and reruns discussed previously.

Another way to tie up an initiator and some datasets in JES2 is through dataset contention. Contention occurs when more than one task needs to use a dataset simultaneously and at least one of them requires exclusive control (DISP = OLD, NEW or MOD). Although TSO will not *initiate* dataset contention by requesting a dataset already in use, a TSO, CICS, IMS/DC or other on-line user may already have a needed dataset open when a batch job begins execution. Many installations have a policy of automatically cancelling any job where dataset contention is reported on the operator console. Again, this can cause the rerun problems identified above, especially when the contention occurs after the first step.

JES3 solves problems associated with dataset contention by not initiating a job until all required datasets are available. But it goes one step further than that. JES3 also prevents all other jobs, both on-line and batch, from using the relevant datasets until the JES3-controlled job is complete. JES3 pre-allocates all datasets before the job begins execution, releasing each after the last job step that requires it.

Dependent job control and deadline scheduling also do their part for productivity by reducing operator intervention and increasing system throughput. A large series of jobs, even a complete night's production, can be scheduled with JES3. Each job will be released automatically by JES3 without operator intervention but dependent upon the successful completion of one or more prerequisite jobs. This eliminates the idle processor time normally lost between jobs; the time required for an operator to notice the completion of one job and key in the command to release the next.

Although much less dramatic but still providing productivity, DITTO has finally been released in a fully supported version for MVS after almost a decade in VSE and VM. However, built into JES3, a subset of DITTO has been quietly supported by IBM for years. For example, *CALL,TT,IN=480,OUT=481 entered from the operator console will copy the first file of one tape to another. To enter the same command through normal job submission (for example, TSO SUBMIT or ISPF Editor SUBMIT), enter //**CALL,TT,IN=480,OUT=481 for immediate execution.

JES2 Advantages

As senior MVS/XA systems programmer for Canada's New Brunswick Provincial Government Data Centre, Cy Schubert has been in both JES2 and JES3 environments. Given the choice, he feels almost any MVS installation is better off with JES3 but identifies three areas in which JES2 does have the edge:

- A better scheduling algorithm in which the priority of a job is boosted by the passage of time rather than whenever the job is bypassed for processing
- SDSF, a popular IBM tool that provides easy on-line access to spool output and resource utilization for both running and spooled jobs, works only in JES2 but software offering some of the equivalent functionality is available for JES3
- IBM's new features generally appear first in JES2.

The last point may make JES2 a better choice for installations deeply committed to IBM's Early Support Program. For example, a new version of MVS may support JES2 before it supports JES3 although IBM has announced simultaneous availability for ESA.

But what about the oft-quoted myth that JES3 is an order of magnitude more difficult to install than JES2? Discussions with a 20-year IBMer who installs both JES2 and JES3 for customers indicate that there is no significant difference in installation time or effort.

JES3 Conversion

So, should you convert from JES2 to JES3? Gartner Group's Michael Braude does not think so. "Unless you have, or plan to have, three or four 'water-cooled engines' (IBM 3090-class CPUs) within the next three or four years, you are better off waiting for Summit."

Commonly called "JES4," IBM plans to supersede both JES2 and JES3 by providing a JES that is upwards-compatible to both, accepting the JCL and console operator commands of both. Interviewed in early 1987, IBM's JES developers stated that they expect to introduce JES4 in 1993. There are conflicting reports from IBM sources as to whether JES will continue to be a separate spooling package or be made an integral part of the MVS operating system. There is also talk of breaking JES4 up into separate functions, some of which will be an integral part of MVS and some that will be provided as additional cost software products.

No matter which JES you may be using, seriously consider reviewing your JCL coding standards to eliminate both JES2 and JES3 statements from new JCL being written or old JCL being maintained.

If you are just planning a new MVS installation, seriously consider JES3 no matter what your size. But if you already have JES2 installed, conversion of all your JCL could probably not be justified by the productivity gains unless you expect rapid growth in a multi-CPU environment. After all, most MVS installations have an extremely large amount of JCL in run streams. In fact, a quick survey of just how much JCL is in your shop could be informative and is guaranteed to surprise you.

However, do not just look at production run streams. Not just every programmer but every TSO user typically has a library of batch jobs that is used every day in the course of his/her duties. The conversion of all this JCL is not hard but it is time-consuming because of the sheer volume involved.

Although conversion to JES3 could rarely be justified on a purely cost-benefit basis, there are some notable exceptions:

- Newer installations that have followed IBM's recommendation to use the OUTPUT and other MVS JCL statements that have been recently introduced to replace most JES2 and JES3 statements found in batch jobs
- Installations using predominantly locally-written or packaged applications that use a JCL generator to produce all run streams
- Installations using predominantly packaged applications in which the vendor-supplied JCL for run streams has been implemented without modification and could be replaced with the vendor's JES3 JCL.

No matter which JES you may be using, seriously consider reviewing your JCL coding standards to eliminate both JES2 and JES3 statements from new JCL being written or old JCL being maintained. As well as saving you a lot of effort if you do convert to JES3 in the future, it could also make conversion to JES4 easier. After all, what happens if JES4 is not quite as upwardly-compatible as everyone thinks it will be? Or if you find you cannot wait for JES4 and you need JES3's capabilities now?

ABOUT THE AUTHOR

Jon E. Pearkins, CSP, CDP is president of Certified Software Specialists Ltd. in Edmonton, Alberta, Canada. He has spent almost all of his 15 years of data processing in technical support both as a hands-on manager and systems programmer. Certified creates and supports its own line of IBM mainframe systems software. Certified Software Specialists Ltd., 54015 Range Road 212, Ardrossan, Alberta, Canada, TOB OEO, (403) 998-0607.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
280 (Interesting/Helpful), 281 (Too Technical), 282 (Too Basic)

2.) Would you like more articles on the same subject?
283 (Yes), 284 (No)

VSAM Optimization and Design

Performance Improvements Under CICS/VS

By Eugene S. Hudders

Even though VSAM was announced almost 15 years ago, it continues to be one of IBM's premier access methods. Organizational options, flexibility and performance are some of the reasons why it is a widely-used access method today.

In the January 1989 issue of MAINFRAME JOURNAL, the effect of proper CISZ selection and the control of CI/CA splits with proper free space specification was presented. In a second article in the February 1989 issue, free space selection, I/O buffer allocation, space allocation and index options available to users were addressed.

As more and more companies depend on telecommunications to resolve daily situations, performance improvements under CICS/VS become a critical area. This article offers possible suggestions for performance improvements and addresses a series of items that, although classified as miscellaneous, may affect the performance of VSAM processing. Most of the miscellaneous items deal with parameter specifications in the VSAM cluster definition.

CICS/VS

VSAM is the preferred access method used in CICS/VS. It provides different access techniques to suit most datasets. The interface between VSAM and CICS/VS is provided through the File Control Table (FCT) and File Control Program (FCP). CICS/VS internal control information related to the dataset is provided in the FCT. Among the parameters are the number of buffers and strings (access paths). In CICS/VS 1.7, the default buffer management is Local Shared Resources (LSR). This is different from previous releases of CICS/VS where the default was Non-Shared Resources (NSR).

Under LSR, buffers allocated are assigned to a pool that will be shared by all datasets associated with this pool. LSR provides an important way to obtain virtual storage constraint relief. LSR provides improved buffer search techniques by looking for the desired data or key in the allocated buffers. This is called look-aside buffer processing. If the desired information is found, physical access to the dataset is eliminated thus improving response. However, increased CPU processing will occur due to this look-aside processing. In releases prior to 1.7, the user had to request LSR by specifying SERVREQ=(SHARE) in the FCT.

In 1.7, CICS/VS will use the FCT parameter STRNO to determine the number of strings and data/index buffers to be allocated to the dataset. Under MVS/XA, CICS/VS will be able to allocate up to eight local resource shared pools. Under MVS/370 and DOS, CICS/VS will only allocate one local resource shared pool. The default pool number is one if not specified.

Under CICS/VS 1.7, the pool(s) is (are) built whenever the first dataset in the pool is opened. If CICS/VS has to compute the pool specifications, it will take time as all the datasets in the pool must be analyzed. I recommend that the user provide the necessary information to build the pool(s) by coding a SHRCTL macro for every pool. This will make the buffer pool allocation faster and will give the user the capability of adding more buffers than the amount computed from the STRNO parameter. In order to get good performance out of the LSR pool(s), the user should ensure that the CISZ for the data and index records (for the datasets participating in a pool) do not coincide in size. This is to avoid having a data request overlay an index buffer of the same size. This will force a read of the index whenever it is needed again. This occurs because LSR uses a Least-Recently-Used (LRU) algorithm in its buffer management. By keeping the data and index CISZ different, there will not be any competition for the same buffer size between the data and the index. In MVS/XA as other pools are available, the user could isolate datasets that may have data and index CISZ conflicts. Also, extra pools provide an opportunity to lower the look-aside processing required by not having to search as many buffers within one pool.

The greatest advantages of LSR are the reduced virtual storage requirements and its look-aside capability. Its disadvantages are the increased CPU processing and that chained operations are not done. Two areas where VSAM will perform chained operations are split processing to format new CAs and browsing operations in order to read ahead. If the user wants better performance for datasets that have many CA splits and/or heavy browsing, then LSR should not be used. Remember that LSR is now the default in CICS/VS 1.7.

The opposite of LSR is Non-Shared Resources (NSR). In this case, buffers reserved for a dataset are for the exclusive

Buffer allocations for CICS/VS should be made in the FCT and not in the JCL.

use of the dataset. The user can provide additional buffers through the BUFNI and BUFND parameters. Additional index buffers will be used to hold the index set indices. Additional data buffers will be used for chained read (browse) operations and for CA split processing. Each string will get one data and index buffer. The first chained request will take up all the remaining unallocated buffers. As the additional buffers are released, the next user can utilize them in a chained operation. Different from LSR, the Sequence Set Index (SSI) and data records are reread. Look-aside occurs for the index set buffers. To obtain NSR under CICS/VS 1.7, the user must specify LSRPOOL = NO.

As datasets are allocated after the CICS/VS start up in 1.7, the user has to revise the OSCOR value. VSAM buffers will go above the 16MB line for MVS/XA but many control blocks must still reside below the line. Care must be taken in this area. LSR should be used whenever possible. CICS/VS shutdown statistics will help identify the look-aside effectiveness. A 70 to 75 percent effectiveness is usually considered good.

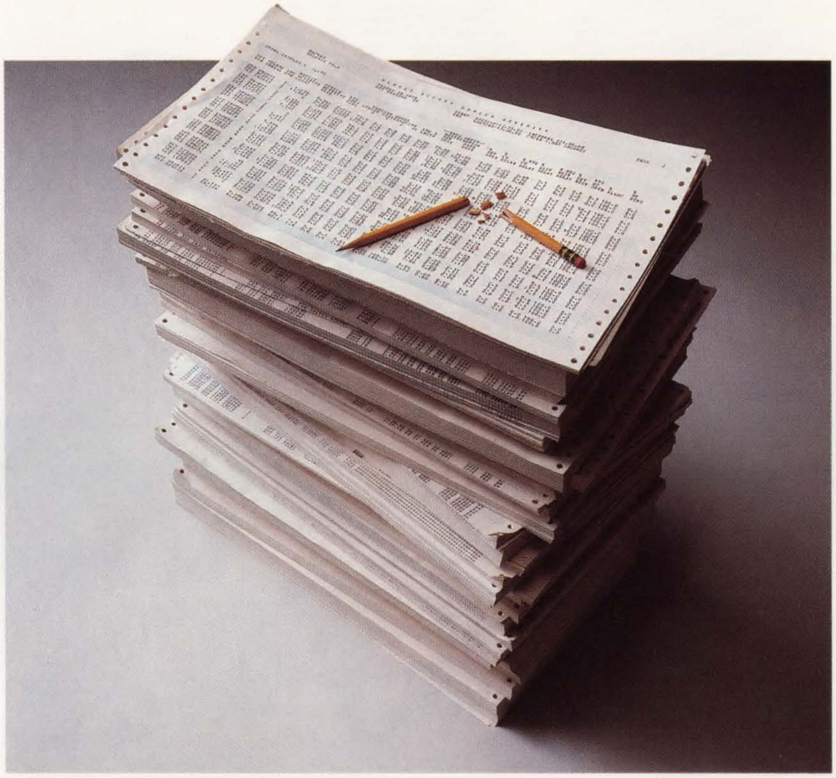
The selection of the STRNO is an important area. Too many strings waste virtual storage while too few affect the dataset performance. A few wait-on-strings are not bad. Monitoring CICS/VS shutdown statistics will help in the adjustment of this value per dataset. A value of three to five is sufficient for most moderately accessed datasets. Heavily accessed datasets can have five to 10 strings while low volume datasets can have one to three strings.

Data and index buffers will vary with the number of strings allocated. The default data buffers are string number plus one (for split processing) and the default index buffers is string number. Additional data buffers can be provided for chained operations using the BUFND parameter. Index buffers can be allocated based on the number of index levels in the dataset (minimum) or the number of index set records plus one using the BUFNI parameter. Both of these recommendations are valid for NSR processing. For LSR processing add these recommendations to the SHRCTL specification. In the SHRCTL macro, the user should specify the total number of strings desired, the total number of individually sized buffers and the maximum key length.

The key length is used in the Place Holder (PLH) control block and must be large enough to hold the largest key. This was a problem in previous releases of CICS/VS when automatically computing the pool if the dataset with the largest key could not be opened. Buffer allocations for CICS/VS should be made in the FCT and not in the JCL. CICS/VS will control strings externally to VSAM so as to avoid region waits.

Other areas that affect CICS/VS performance are protected resources, browsing, split processing and share options. Protected resources are datasets for which Dynamic Transaction Backouts (DTB) are to

MOST DASD PERFORMANCE SOFTWARE TELLS YOU WHAT'S GOING WRONG.



WE TELL YOU HOW TO MAKE IT RIGHT.

Until now, DASD tuning was a painstaking task-requiring hours of statistical analysis just to figure out what was wrong. Correcting problems often involved guesswork, trial and error and just plain luck. DASD ADVISOR from Boole & Babbage tells you exactly what's going wrong and how to make it right.

DASD ADVISOR is an EXPERT system-based DASD tuning tool that eliminates the need to wade through piles of performance statistics. It analyzes the performance of your entire DASD subsystem, from individual data sets, through hundreds of devices, controllers and channels. It identifies data bottlenecks, then makes specific tuning recommendations. All in concise English. So you have what you need to improve DASD performance. And time to solve other system performance problems.

For a free demo diskette that shows you how DASD ADVISOR can help *you* make it right, call Marty Johnson. In California: 800-624-5566. Outside California: 800-822-6653.

Boole & Babbage, Inc., 510 Oakmead Parkway, Sunnyvale, California 94086.

Boole & Babbage

The Performance People

CIRCLE #165 on Reader Service Card ▲

occur. If a task abends before a Syncpoint is taken, all updates are reversed and any updated records are restored to their original status prior to the updates. In order to accomplish this, CICS/VS must maintain control of the updated data records even after the updates are complete. There are two methods of releasing this control that are at task end (implicit) or by issuing a Syncpoint command (explicit). If the user can logically issue a Syncpoint at some point in the program without endangering integrity, then it should be done in order to relieve any potential contention for the record. If this explicit command is not issued, the records will be unavailable to other tasks until the current task ends and an implicit Syncpoint is taken.

Browsing is nothing more than providing the sequential reading of the dataset starting at a particular point. Unfortunately, it is an abused feature that can adversely affect performance as many resources can be consumed especially if additional data buffers are available. A lot of data can be read tying up the device, control unit and channel. This data may not be used. Due to this, I recommend placing browse datasets into an LSR pool to avoid reading ahead and tying up additional resources. A better alternative is to place browse datasets into their own separate pool to reduce flushing buffers. It is also recommended that programmers only read sufficient data until a screen is full. Stop at that point, save the screen on auxiliary temporary storage for future reference, display the screen so the operator can see if the browse has been satisfied and save the last key displayed in case you must continue the browse. This technique will tend to smooth out response time and tie up less resources at one time.

Split processing takes time, especially if a CA split is occurring. Splits affect response time. Sufficient free space should be allocated to reduce the number of splits that occur while processing on-line. CA splits can take seconds to complete and should be monitored. Also, during CA split processing, the dataset is unavailable to other users until the split is completed. Reorganizations may be required in order to again acquire unusable free space.

VSAM provides many share options. Avoid specifying unnecessarily SHAREOPTIONS 4 (cross region or cross system). With this option, the buffers will be refreshed for every I/O causing a degradation in performance.

One final comment regarding CICS/VS deals with a testing environment. Test systems do not need to have a large number of strings and/or buffers allocated. In a test environment, there is a limited number of terminals accessing the data. Too many strings and/or buffers can be a waste of resources. So, use only one string and default the buffers. This will also help in locating programs that hold more than one string since the task will hang due to lack of strings.

Miscellaneous

There are certain miscellaneous items that can be done to provide improvements to VSAM KSDS file. The specification of SPEED in the cluster definition is one. The default is RECOVERY. This parameter (SPEED) can improve dataset load times by reducing the amount of overhead caused by the preformatting of Control Areas (RECOVERY). Once the dataset is loaded, RECOVERY will be in effect. RECOVERY is always in effect for the index portion. If the dataset is large enough, the preformatting may affect the total load time.

Another parameter that is sometimes specified is ERASE. This parameter requests that the area occupied by the cluster be reset to binary zeroes whenever the dataset is deleted. If the dataset is large and is deleted often, a lot of time can be wasted by unnecessarily specifying this parameter. This parameter should only be used for high security datasets and definitely not for every dataset. Protection from unauthorized access should be done through a security package.

DASD reliability has improved tremendously over the past twenty years. This reliability has more than nullified the need to issue write checks after each write for data verification. The mechanism used to verify a write on disk is to wait a full revolution to read the data that was just written. This will add an extra 16.7ms on a 3380 DASD unit to the access time. In some cases this can double the response time. The verify read does not actually read the data into storage since the skip bit (suppress transfer) in the CCW is on. In this manner, the data is read and Error Correction Code is regenerated and checked but the data does not get to central storage. It is important not to specify WRITECHECK in the cluster definition. If the dataset is such that the user has to be sure that the data is correct, then use a mirror copy on a separate disk. This will provide better protection against other types of data losses (for example, head crash, inadvertent overlay, DASD malfunction and so on) which the verify option does not include. If the mirrored dataset is placed on a different path, then a certain amount of overlap can be provided.

One final area where care should be taken is the specification of the Control Interval Size. The CISZ should be specified at the data level and at the index level if the user wants to override the VSAM selection. If the CISZ is specified at the cluster level, then the CISZ will apply to both the data and the index. This may result in a larger than necessary CISZ for the index wasting virtual storage space.

Conclusion

There are several software packages available on the market that can help improve performance and the analysis of the dataset. It is important that the user understands the basics of VSAM tuning so that the recommendations made by these software packages can be properly implemented. Packages do not necessarily correlate different data related to the user environment. For example, some recommendations may be good from a performance point of view but may adversely affect the DASD space utilization that may be your installation's weak area. So, knowledge of VSAM tuning will help in making the right decisions.

ABOUT THE AUTHOR

Eugene S. Hudders is president of Multiple Computer Services, Inc. (MCS). In 1983, he became a Certified Data Processor (CDP) and Certified Computer Programmer (CCP). He is also a Certified Systems Professional (CSP). During 15 years at IBM, he was a systems engineer, SE manager, marketing manager and branch manager. He has also been general manager for MAI del Caribe. Multiple Computer Services, Inc., G.P.O. Box 1697, San Juan, PR 00936-1697, (809) 721-7200.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
285 (Interesting/Helpful), 286 (Too Technical), 287 (Too Basic)

2.) Would you like more articles on the same subject?
288 (Yes), 289 (No)

What Programmers Don't Know

And Why They Don't Know It

By Harvey Bookman

Desperate at three in the afternoon, the project manager needed a knowledgeable COBOL programmer. A report program was not working and none of the more than 20 programmers on the large mainframe project had any idea how to fix it. The client had been promised the report would be in his office two days ago. Although the program coding had been completed in time, the last two days had been spent attempting to resolve an operation exception (OC1) occurring every time the program executed. Pressure had been applied from the client to the vice president who paces through the programming area trying to ascertain the status of the report program. Threats have been made by the vice president to the project manager who, after running out of threats to his programmers, called me.

I was informed that a dump occurred when the program had finished processing and was attempting to return to the operating system with the COBOL 'STOP RUN' command. The programmers were stumped because completely erroneous data got into the registers and the next machine instruction to be executed was at a bewildering location.

Aware that a program must reload the registers upon completion and that IBM VS COBOL saves the registers in the beginning of the Task Global Table (TGT), I suspected that the beginning of the TGT had been overlaid. Furthermore, since the TGT is located directly after WORKING-STORAGE in IBM VS COBOL, I suggested that an invalid subscript, exceeding the maximum size of a table, had been used to reference and store data. There was a good chance that the table might be the last one in WORKING-STORAGE and that the values in the registers might be data that belonged in the table. My supposition proved correct. Within a few hours the problem was solved and the report was produced the following day.

Although the problem depicted above was not an easy one, it was by no means an insurmountable one. Why is it that more than 80 percent of the IBM COBOL programmers have little or no knowledge of the information contained in the most important debugging control block of COBOL, the TGT? Why do many programmers lack valuable knowledge in so many useful areas, consequently causing lost productivity and unnecessary expenses to their companies?

What is the key factor that gives one programmer a vital edge over another programmer? Could it be that industry and management have grouped different types of personnel under the same classification and title, namely *computer programmer?* Might industry be pushing computer personnel toward becoming the less productive type of programmer? If this is so, might not the mere recognition of this fact save untold dollars?

Some might argue that I.Q. is the distinguishing factor here. Others may say that many computer programmers are not interested in their profession, working in their current position simply because it is an excellent base from which to advance into corporate management. Although these are considerations, they alone cannot begin to account for the relatively poor showing of COBOL knowledge that was statistically verified through the use of TECKCHEK (an expert system designed to serve as a standardized instrument for gauging programming proficiency).

A case in point concerns one of the most basic skills of programming, the moving of data from one place in memory to another. While it would seem that proficiency in this area should be extremely high, less than 25 percent of the individuals tested revealed an understanding of data manipulation. Figure 1 shows a question in this category. (The length of the fields and their values have been slightly changed but the question's concept is exactly the same as in the actual test. This example and others used in this article are from the IBM VS COBOL Application Programming Test.) Only 46 percent of the individuals faced with this question were able to answer it correctly. Four percent chose to skip it, admitting that they had no idea of the correct answer. Thirty-three percent of the respondents chose the answer "E," that indicates a lack of the conceptual knowledge that the alignment of a decimal point is done by the COBOL compiler. Technical managers would certainly agree that a programmer should have knowledge of this basic fact.

FIGURE 1

What will be the result of executing the highlighted code?

```
DATA DIVISION.
WORKING-STORAGE SECTION.
77   FIELD1       PIC S9(5) COMP-3 VALUE 12345.
77   FIELD2       PIC ZZZ.99.
PROCEDURE DIVISION.
     MOVE FIELD1 TO FIELD2.
```

A) The program will get an 'E' level error message during compilation.
B) An overflow program check (ABEND) will occur.
C) FIELD2 will be equal to 123.00
D) FIELD2 will be equal to 345.00
E) FIELD2 will be equal to 123.45

Another case in point is exemplified by Figure 2. This question, nearly identical to one posed by TECKCHEK, is designed to determine whether one knows what a binary number is and is able to relate that understanding to function with the binary number format specific to COBOL. Since the binary numbering system is used as the foundation of all computers, it would seem likely and reasonable to assume that nearly 100 percent of those responding would answer the question correctly. Surprisingly, the statistics show that almost 60 percent of the respondents answered the question incorrectly.

The years of productivity lost due to inadequate familiarity with the elements of any given programming language defies imagination.

Although one may argue as to whether or not it is necessary for a COBOL programmer to be adept with the knowledge and understanding to use binary numbers, one would be wise to consider that binary fields are requisite for dump reading, in particular to evaluate COMP fields (widely used since it is the most efficient subscript data format), as well as indices (that are always stored in binary format by the COBOL compiler), displacements in the data map (used to locate fields in a program) and any virtual storage addresses.

So what does a programmer do when he has to determine the value of a binary field in a dump? — Do not reply too quickly! Although there are a number of good dump analysis tools available, they are not going to convert all values of subscripts and indices into decimal. One is faced with the choice of either learning to use binary numbers or coming up with a less efficient and less eloquent method. Statistics indicate that the second approach is more widely used.

When a programmer has a problem and gets a dump, rather than carefully analyzing the dump, he or she will frequently put DISPLAY statements into the program. These are used to print out data, often numeric, so that the values of the fields possibly causing the error can be examined. (A simple question on the DISPLAY verb was answered correctly by more than 86 percent of programmers tested.) After updating the program, it is recompiled, relinkedited and rerun before the displayed information is used to begin the problem analysis. This is an extremely inefficient and sometimes an illogical way to proceed. The problem analysis can begin immediately when the dump is received and the dump will often contain more information than will be obtained by a subsequent run with DISPLAY.

The years of productivity lost due to inadequate familiarity with the elements of any given programming language defies imagination. I can still remember one programmer asking for help after she had spent more than five weeks attempting to determine whether a two-byte field defined as packed decimal (COMP-3), contained a hexadecimal '000C' or '000D'. Each employee record that a client had sent to her company contained a field with this definition. The client wanted different reporting dependent upon this field, the specifications describing it as having a value of positive or negative zero. Had the field values been looked upon as a binary number the problem would have been resolved in five minutes, NOT FIVE WEEKS!

Obviously there are a number of exceptionally good programmers but as any project manager will concur, their percentage is small compared to the number of programmers. It can be seen that many,

FIGURE 2

What is the decimal value of FIELD1 if its hexadecimal value is '002F'?

```
DATA DIVISION.
WORKING-STORAGE SECTION.
77   FIELD1       PIC S9(4) COMP.
```

A) 2
B) 17
C) 35
D) 47
E) None of the above.

if not most, programmers will attempt to accomplish a given task with the least amount of work (which is good) and the least amount of knowledge (which is bad). As a means to exemplify that programmers often have only the minimal knowledge to do their job, consider the response to a question concerning which COBOL fields can be checked for numeric. While almost 90 percent had partial knowledge of doing numeric testing, less than four percent of those demonstrated awareness that COBOL allows the test on alphanumeric display, numeric display and numeric COMP-3 fields but does not allow it on numeric COMP fields. Fifty-nine percent of those tested were aware that it is allowed on alphanumeric display fields, 67 percent that it is allowed on display numeric fields while only 32 percent knew that it is not allowed on a numeric COMP field.

What makes the statistics so alarming is that the statistical data employed in this article has been compiled from a wide range of professional programmers representing a broad sector of the country, including programmers from entry level to those with more than 15 years of experience. So we are still perturbed by the following questions: Wherein lies the difference between one programmer and another programmer? Why are there such a large number of computer personnel without enough basic knowledge to work effectively and efficiently?

Few would argue that high productivity is a useful asset valued within many areas of life and is the bedrock in any corporate structure. Many seasoned corporations spend much energy and money on motivating personnel to produce at their maximum potential. Methods ranging from vigorous internal competition to almost excessive rewards signify the value placed on productivity. However, perhaps more due to the demand for ever increasing impressive results, many corporations and their management personnel have been led into a heavy-handed approach with regard to getting the job done. In so doing, they fail to recognize that results are not only a matter of degree but also a matter of quality. Although output is increased, they fail to take into account all of the negative aspects of their approach.

Now if we consider this a little further, it might become clear that although increased productivity can lead to financial gain, by its very nature it cannot serve to distinguish between the "drive to understand it and do it right" and the "drive to get it done at any cost." Although high productivity may seem to be the most favorable end, it may force sacrifices that in the long run will prove to far exceed any gain. We find ourselves wondering, "What has been sacrificed?"

Sacrificed are efficient and maintainable programs. Not taking the time to carefully think out each step before proceeding frequently produces sloppy code. However, the biggest sacrifice is that the programmers who worked on the project, constantly going onward without pause for contemplation, have remained in a stagnant position. They have added little to their proficiency and understanding to aid them in accomplishing future projects. Programmers who are driven from purely financial motivation, worrying solely about productivity, often will not forego the added expense of investigation that would bring art, which is the most efficient form of data expression, into the design of programs. On the other hand, those programmers who from the outset are motivated by the art they encounter in programming will spare little to discover the many facets of their craft and in so doing will, like the old masters and their respective art forms, be ready with this new age art form with tools in hand, purpose in mind and care in heart

ABOUT THE AUTHOR

Harvey Bookman is president of Bookman Consulting, Inc., a software development company specializing in programmer proficiency testing. Bookman is responsible for the development of the expert system testing service, TECKCHEK. Bookman Consulting, Inc., 67 Wall Street, Suite 2411, New York, NY 10005, (212) 819-1955.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
290 (Interesting/Helpful), 291 (Too Technical), 292 (Too Basic)

2.) Would you like more articles on the same subject?
293 (Yes), 294 (No)

Trax Softworks, Inc.

Product Support Is Big Business at Trax Softworks, Inc.

From the fifth floor of a six-story office building in West Los Angeles, Trax Softworks, Inc. produces some of the most powerful end-user mainframe software in the industry. They have a worldwide customer base with 25 percent of their unit sales outside of North America. Their mainframe spreadsheet program, Electronic Spread Sheet (ESS) is translated into Spanish, French, German and soon-to-be Swedish. With so many users here in the United States and throughout the world, there is a definite need for *quality support*.

Quality Support

Trax Softworks was founded in 1981 by Leonard Fischer and Dwight Harm. They were working for Western Airlines in Los Angeles as the IBM mainframe MVS and VM systems programmers. Fischer's early interest in phone company telecommunications hardware led him to pursue a way to access dial-up database services. Although these services were widely available, they were not accessible by an IBM mainframe computer. While the co-founders were discussing possible software solutions to this problem, Trax Softworks came into being.

Since 1983, Trax Softworks, Inc. has designed its products to make the already hectic lives of the mainframe users easier. They began with a simple three-point *support* strategy:

1) Target the software applications to the end users who may not have extensive computer systems background
2) Design products that solve business problems
3) Provide quality support to maintain the products and vendor-client relationships.

Adherence to these fundamentals may be one explanation for Trax's survival in the highly competitive computer software world. The company has not allowed anything to let it stray from its original philosophy of providing end users with a product that is not only easy to use, but also powerful, flexible and efficient. All of Trax works in unison to perpetuate this ideal.

Trax software applications are developed with the end user in mind. By creating programs that run on the mainframe and are user-friendly, Trax helps to dispel the myth that only systems programmers can run mainframe software. With Trax products the end user holds the power of the mainframe and the user-friendliness of a PC.

Due in part to its small size and open management style, Trax can react quickly to changes in the mainframe marketplace. When the subject of IBM's new small business-targeted, mid-range 9370 is mentioned, F. Thomas Cox, vice president of marketing, responds by saying, "The trade press keeps telling us about the dearth of software for the 9370s but all of our products run exceptionally well on them as well as on other IBM mainframes and compatibles. We've got it all."

Problem solving is another motivation behind Trax products. Everything is done with solutions in mind — specifically business solutions. Most product enhancements come from customer suggestions. In ESS, the SQL/DS and DB2 database interfaces, Lotus macro compatibility and graphics program interfaces have all come from suggestions. Trax's customers have come to expect quick and effective results.

It is only through constant interaction with clients that Trax can maintain such a high standard of support and service. The heart and soul of product support is the hotline. It is the direct link to the end users and the place where vendor-client relationships are won and lost. Leslie Thomas, manager of support services, says, "We try to give the customer that extra service above and beyond a simple answer. Giving the users the 'why' and 'how' helps them to increase their own productivity." This kind of personalized service enhances the value of the product.

Contrary to belief, product support is not just operating a hotline. Other areas of responsibility include quality assurance (software testing), user training, presales demonstrations, technical documentation, a quarterly newsletter and an annual user group conference. Product support specialists must be able to wear different hats in different situations.

The Trax Product Line

The first product to bear the Trax name was VM DialOut. It was originally conceived as a solution to the problem of using a "clunky," acoustically coupled terminal to access the on-line VM Share bulletin board. VM DialOut allows users at 3270 terminals to access systems that require asynchronous terminals. The advantage of TSF is that a user has one terminal to access networks like TYMNET and The Source and can use that same terminal to access his or her IBM mainframe as well as other systems such as VAX, Hewlett Packard, Prime or UNISYS.

ESS was one of the first spreadsheet programs for the mainframe. ESS hit the market in early 1983 and has become the product with which Trax is most readily associated. The current version, which is panel driven, can read SQL/DS and DB2 tables and execute macros written in the Lotus in-spreadsheet macro language. ESS was also one of the first mainframe spreadsheets to read and write Lotus binary worksheet files.

Trax's answer to word processing for the mainframe is EdWord. There is no need to memorize complex formatting codes or commands with this PC-like word processor. EdWord has pull-down menus, formatting rulers, a spelling checker, multiple windows and a print previewing feature. An ESS-integrated version of EdWord is available giving users the combined power of the spreadsheet and the word processor.

Trax's latest product, TopNotch, is a mainframe version of electronic desktop managers originally seen on PCs. Individual windows for each of the five functions can be moved, resized and shuffled. There is a mini-spreadsheet called a "spreadpad;" a database-type index card filing system; an appointment calendar with hourly, daily and monthly pop-up alarms; a mini-word processor called a "notepad;" and a toolbox for printing and transferring data displayed on the screen to disk or paper.

Back to Basics

Trax Softworks, Inc. is a company that has not lost sight of its roots. It is the end users who propelled Trax into today's market and it is the end users who will continue to guide the company into tomorrow. Its users have come to expect quality products as well as quality support from this constantly growing company with a bright future.

Vendor Profile is a regular forum whereby a vendor is given the opportunity to introduce the company and its products to MAINFRAME JOURNAL readers.

TECH ADVISOR

Questions are answered by consultants and instructors from Davis, Thomas & Associates (Minneapolis, MN), the largest technical services firm in the upper midwest. Please address your technical questions to: The Tech Advisor, MAINFRAME JOURNAL, PO Box 551628, Dallas, TX 75355-1628.

Q *Our installation has many high-activity on-line (CICS 1.6) VSAM files with alternate indexes that are also accessed on-line. It appears that to allow CICS to access either the cluster or AIXs at the same time, everything must be defined as share-option 4. If the share-option 2 is used, the cluster will open to CICS but the AIXs get open errors. Is there a way around this or must we use share-option 4? If so, can we put these clusters and AIXs in the LSR pool even if the AIXs are in the upgrade set? If not, what can be done to minimize the number of physical disk accesses against these files?*

A I am assuming from the nature of your questions that you are a VSE shop and that the alternate indexes in question are upgrade datasets.

First, the question about the use of share-option 2 versus share-option 4. It would seem logical that you must use share-option 4 because you are basically opening the same dataset twice in the same partition. When CICS goes through the FCT opening the files, it will open your base cluster as you specified it to be opened in the FCT. It will then continue on through the FCT until it reaches your alternate index entry. CICS will now try and open the alternate index and the base cluster. This will cause an open error since the base cluster is already open. Your only option is to code share-options 3 or 4 on the VSAM definitions and let VSAM handle the file integrity.

As for putting the entries in the LSR pool, the *CICS Performance Guide* (SC33-0134-1, page 190) states that CICS automatically excludes alternate indexes and base clusters with upgrade sets from the LSR pool. This changes when you are running CICS 1.6.1. but only for CICS/OS. With OS they have added a parameter in the DFHFCT TYPE=DATASET called BASE that describes to CICS the base dataset of an alternate index structure. It is coded on the alternate index FCT entry. This provides read integrity between the alternate index paths and the base dataset along with saving virtual storage since everything will be routed through one set of VSAM control blocks.

I am afraid there is not much you can do regarding the number of physical disk accesses you have when coding share-option 4. The accesses you see are being generated by VSAM to guarantee read/write integrity and you have no control over how this is done.

Q *What problems are there, if any, in processing a VSAM alternate index natively, rather than through the path for inquiry purposes only. Our shop has been doing this for as long as we have had VSAM files and we have not experienced any corrupted alternate indexes. We are aware of the potential inaccuracies of the data due to timing but feel the processing results are worth the risk. Reading natively has reduced processing time by as much as 75 percent in batch programs. We tried the textbook method of accessing through the path but this resulted in lengthened processing and on-line response time. I emphasize that this is inquiring through the alternate index and not updating. We have always updated through the base cluster. I have asked this question because most people outside our shop have never heard of this method. Have we just been lucky for these many years?*

A There is nothing wrong with reading the alternate indexes yourself. If all you are doing is verifying keys or picking up some other information contained in the key you are safe. You have eliminated extra reads to the base cluster saving yourself significant time and I/O.

From the tone of your question you understand what the potential risks are and obviously have found operational procedures to compensate for them. Your method of updating only through the base cluster is highly recommended for integrity reasons. If you decide to make your alternate indexes upgrade sets this would add a fair amount of overhead into your system and you seem to be concerned already with response time and batch throughput.

The alternate index could even be updated by your own programs if you desired but I would *never* recommend doing this. You would have to have some very good procedures in place to maintain the integrity of the files. As you already know, alternate indexes are just VSAM files built via an IBM utility from information gathered from other VSAM files.

Q *How do you recover user data from CICS journals, including DL/1 data, not using IBM IMS recovery techniques?*

A This is one of those famous ''loaded questions.'' From the information I can uncover, you have three basic options available to you and each one gets fairly complex. You can either use IBM's IMS recovery techniques, buy a third-party software package that does this or write your own system. The last option would not be one that I recommend because of the tremendous amount of time and effort you have to put into developing it. This is not to say it cannot be done but in the long run the maintenance costs could be more than the other options.

By far the simplest approach would be to do some research and find a product that meets your recovery needs. A reference book such as *Data Sources* should be able to give you several leads. If you are a member of any user groups, I would ask around at these meetings and find out what other companies are using. This could save you a lot of time by getting opinions from actual users.

Finally, you could spend some time with the IBM manuals and learn IBM DBRC. If you decide you do not want to use IBM's recovery techniques, you could ask your IBM representative if he knows of different solutions to your problem. He may have access to other products that could help you or be able to point you in the right direction. At the very least he should listen to your concerns and relay them through the proper channels to the support team.

Q *What are the advantages/disadvantages of routing dial-up communications through a local 3x74 versus the front end?*

A A disadvantage of using a 3x74 is that all error recovery is done by the modems. There are error detecting modems available that are slightly more expensive than ordinary modems. These modems will help reduce line error problems. Another disadvantage of an ASCII device is that each time a character is typed, it is also transmitted down the line (watch the send data light on your modem when you depress a key). With slower modems this causes a lot of key stroke errors. The 3x74s do not have ASCII ports so a black box is also required to change the ASCII protocol into a 3270 datastream. The advantage of ASCII is that there is less need for error recovery and the cost is quite a bit less than a 37XX.

An advantage of the 37XX front end is that you have cards/software at the remote end that make your equipment SNA and let the NCP/VTAM do the line protocol conversion and error detection. This disadvantage of a front end is the cost of both the hardware and the software to support the hardware.

Q *What is the ideal block size for any mainframe file; for both performance and efficient DASD utilization?*

A Everyone will have a different opinion on the ideal block size for any file. It is like asking what is the ideal temperature or the best car. Each person you ask will tell you what he likes best based on the environment. This question may be answered with another question like, "Under what conditions?" or "What are you trying to accomplish?" In short, you will find what is good in one situation could be bad in the next.

When looking for a block size standard in your shop, you should develop it so it outlines for your staff how to choose a block size rather than setting one rigid number that everyone must follow. In each situation the needs will be different. One time you may need optimum DASD utilization and the next time fast batch through and in the next case efficient use of virtual storage will be the overriding issue. Develop a standard that helps people ask the correct questions and find the best answers for their situations rather than dictate a block size for all files. I know it would make life simpler if we set such a standard but, unfortunately, life in DP is full of compromises.

There are some basic questions that will need to be asked when choosing a block size. Some of them are:

- What type of access is used most often with this file, random or sequential?
- What is the record size?
- What is the file organization, sequential or keyed?
- What is the file access method, VSAM, ISAM, BDAM?
- Will the files be used in an on-line or batch system?
- What type of DASD are you using?
- What is the track size of the DASD?
- Does my system have enough virtual storage to handle large I/O areas?

There may be more questions that need to be asked based on specific conditions in your shop. However, those listed above should be asked every time you choose a file block size.

After you have answered these questions, then you can choose a block size based on the answers you received. A simple set of recommendations might be the following.

- Sequential file processing — full-track or half-track block sizes. This will give you high DASD utilization, fewer I/Os but will take more virtual storage for I/O buffers. If the file is a VSAM file, then define the CI Size as the MAX-CI available for your device type. When you are using a device with a large track capacity such as a 3380, then use half-track blocking to reduce virtual storage usage.
- Random file processing — I would recommend block sizes large enough to contain a few records but not much larger than 4K in size. This is especially true with on-line files since in most cases you will be working with only one record at a time anyway. With random processing, your biggest concern is how the application program is coded and how it uses the file. A poor application design could harm you more than having a bad block size on the file.

Remember that with most tuning activities you are constantly compromising. Unfortunately, there is no set rule that says this is good and this is bad. If you would like more information on using your DASD efficiently, I suggest you read "Water On The Rock" by R. Douglas Swords, July/August 1988 and/or "VSAM Specification and Tuning" by Frank Bereznay, May/June 1988. These articles have helpful information on DASD use.

CIRCLE #129 on Reader Service Card ▲

Multiple Multiple Multiple Multiple Address Spaces In VSE

VAE is a great idea, especially for shops that have outgrown a 16MB single space system.

By D. Robert Bailey

An important consideration in migrating to VSE 2.1 and higher is the implementation and use of Virtual Addressing Extensions (VAE) storage. This is a new and sometimes confusing concept to systems users who have previously had no exposure to multiple address spaces. Multiple address spaces and its architecture in VSE need not be a stumbling block for users if presented in its basic form.

This article offers some insight into the organization and mechanics of VAE for VSE. Follow the evolution of address spaces from simple virtual storage through multiple address spaces. Along the way, some important side issues will be addressed.

Beginning At Location Zero

In the beginning IBM created real storage and the user said, ''This is good! But could we have a little more?'' And so it began, a never ending cycle of more and more storage requirements: yesterday's abundance seems like a trifle today. There is never enough!

The response, of course, was the idea of virtual storage. Why hog real storage if you were not using it? The problem with this idea was stumbling up against the 16MB limit of the 24-bit addressing scheme. The response was in two dimensions: Extended Addressing (XA) and multiple address spaces. XA is not available in the VSE world but multiple address spaces are.

One Address Space

Before looking at multiple address spaces, an understanding of how a single virtual address space is organized is in order. The 370 architecture expects two tables to tell the system which real page represents which virtual page. The highest level table is the segment table; each entry in the segment table represents 64K of storage by pointing to a group of page table entries. Each page table entry represents 4K of storage. Tell the system where to find the segment and page tables. The hardware handles where things are and when to page in or out.

A non-VAE system with 16MB of storage is organized as shown in Figure 1. The supervisor resides in ''low core'' followed by allocated problem partitions; the Shared Virtual Area (SVA) is located at the high end of storage. The supervisor and the SVA are allocated with a Protection Key of zero while problem partitions are allocated with a Protection Key equal

FIGURE 1

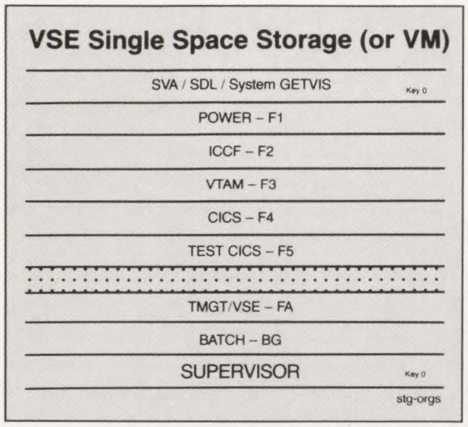
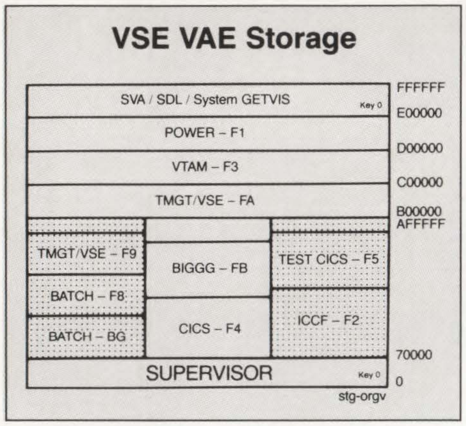


FIGURE 2



to its Partition Identification Key (PIK).

The hardware translates address references by using information found in the segment and page tables. If the page is already allocated, the operation proceeds. If the referenced address belongs to a page that is currently out, a page fault occurs and VSE reads the page into real storage from the Page Dataset (PDS).

When 16MB Is Not Enough

When IBM announced virtual storage with SIXTEEN MEGABYTES! "all us poor programmers" thought we had died and gone to storage heaven. We thought all that storage was just for us. But it is amazing how fast a dozen or more competing jobs can use up "all that storage." Soon DP managers everywhere were complaining to their bosses and to IBM, "We have used up all of our virtual storage which really doesn't exist but we can't buy any more!"

After IBM had heard this for a while and had given up on thinking about any more vertical growth, some bright engineer started thinking horizontally (or sideways) and conceived the idea of multiple address spaces.

Multiple Address Spaces

If one table can be used to create one address space, then why not use two tables to create two address spaces? Of course, there are a few minor problems: are there two supervisors? what about shared modules? how does the system know which table to use? who owns the I/O devices? . . . Little problems like that.

As it turns out, however, the problem has been solved by using shared partitions and by making the supervisor and the SVA areas common to all address spaces. The organization of a three address space VAE system is shown in Figure 2. Notice that the supervisor, POWER, VTAM and the SVA stretch all the way across and are always available to all partitions. It should not be a difficult transition to think of this system as three separate systems with only one running at any one time.

Implementation of these multiple address spaces is accomplished in just that way. The segment table is replicated to map three address spaces and the page table is expanded to allow for 40MB as shown in Figure 3. The portions of the segment tables for all address spaces that

Although IBM now supports up to nine address spaces in a system (thanks in no small measure to Pete Clark), adding address spaces contributes to overhead.

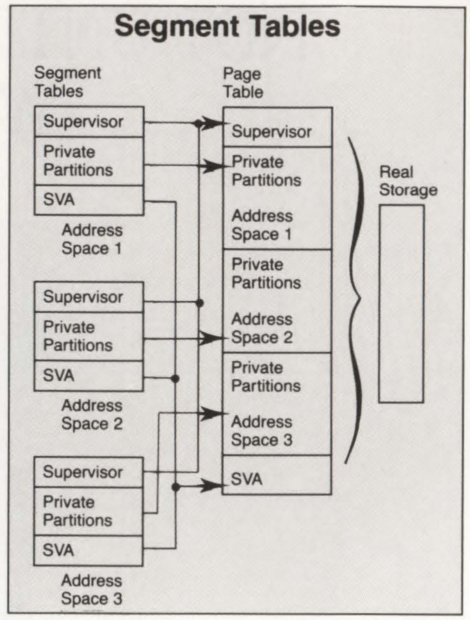
refer to supervisor, SVA or shared partitions will point to the same page table entries. The non-shared or private portions of the segment table will point to unique portions of the page table.

The effective real address of any storage will be:

00 SS P BBB where

SS comes from the segment table, P comes from the page table, and BBB comes from the original virtual address.

FIGURE 3



Other Points

All system control blocks reside in shared areas: either the supervisor or the SVA. This is necessary since an I/O or page-in can complete at any time whether the address space is active or not. Also, there is no "swapping out" of address spaces as there is in MVS. All pages for a particular address space could eventually be paged out if all of its partitions are inactive but there is no conscious effort on the part of the system to do this.

Observe that for a given number of address spaces in a given system, shared areas come at the expense of non-shared areas. Although IBM now supports up to nine address spaces in a system (thanks in no small measure to Pete Clark of Olan Mills), adding address spaces contributes to overhead. Nothing is free!

You should also pay attention to the high likelihood that paging rate will increase. By how much depends on how much real storage is already in your system and how much paging was going on beforehand. If paging becomes a problem, IBM will be only too happy to sell you more storage. As I said, nothing is free.

One final point. Although it is possible to implement VAE on a VM guest machine, this is not a good idea. With a single space 16MB guest, VM handles all the paging; for a VAE guest, VM *and* the guest handle paging. In effect, there can be double paging. In any event, there is only one VAE guest allowed per VM system.

See VSE *page 98*

CIRCLE #66 on Reader Service Card ▲

Solid State *from page 63*

full platter rotation so that the head is again positioned correctly over the data. This process takes 16 ms for an IBM 3380 type disk during which time an IBM 3090 Model 200E could have processed about 800,000 instructions had the data been available.

One other common approach is to overallocate disk space dedicated to paging. By overallocating paging space that is used to store data temporarily while the CPU is performing work with other data, unused disk space is tolerated in an attempt to improve system performance. Along with the overallocation of paging space comes an overallocation of channels dedicated to that paging function. It is not uncommon to find a site that has dedicated more than 1,000MB of storage (an entire 3380 drive) to the paging function with only 10 to 20 percent of that storage actually actively being used for paging.

All of these techniques require tremendous resources of either hardware or programming time to be successful. And while they can help to shorten the performance gap between disk and CPU, they cannot eliminate it.

Solid State Storage to the Rescue

Solid state storage aids the performance analyst in several ways. Typically, about 50 percent of the requests for data made by the CPU are made for the same three to five percent of the site's data (for example, database indices or system files). This phenomenon defeats many of the DASD tuning techniques because it demands repetitive requests to the same channels and drive units, thus building up the queues that result in slow performance.

However, SSD does not allow request queues to build up because of its fast access time. Thus the performance analyst is helped with hardware that can sustain the performance requirements of the most accessed data.

In this application, the SSD functions as a large cache device with 100 percent correct guesses. Users do not need to have all their data on SSD but only the most highly active data.

Because of their fast service times (as low as 1.5 to 2 ms compared with 25 to 30 ms for rotating disk) users can use 100 percent of the capacity of the solid state device. Thus a 1,000MB solid state device will be able to replace four or five 3380 devices that are only partially utilized for performance reasons. This advantage significantly decreases the cost differential between the two media.. The drives thus replaced can then be used for less performance-critical applications in which their full capacity can be utilized.

In highly random, interactive environments such as a transaction processing system, SSD can be effective as well. Because of the higher storage capacities now available on SSD, users can put entire databases on SSD for maximum performance. Also solid state disks now employ back-up systems using combinations of battery power and compact Winchester disks. In the event of a power-out loss, the entire contents of the SSD are backed up to the Winchester disk. This eliminates the chance of data loss previously associated with volatile solid state storage.

In short, solid state devices eliminate the need for inherently slow mechanical devices in the on-line data storage system. As a part of the total storage system, they help solve a vast majority of the disk tuning problems plaguing systems analysts and performance specialists. The SSD of today is to the computer industry as the turbo charged, high performance automobile is to the transportation industry. Each has its place as a performance machine and the price performance of each is likely to decrease with new technological innovations.

ABOUT THE AUTHORS

Dennis Corburn is manager of Systems Engineering at EMC Corporation. Hopkinton, MA. He works with EMC's mainframe products group to determine accurate performance and storage solutions for mainframe users. He has more than 20 years of experience in the computer industry including work with IBM and National Advanced Systems.

Brian Fitzgerald is manager of Marketing Communications at EMC. He is responsible for EMC's corporate communications and technical documentation. Brian's technical writing background includes topics such as caching, DASD tuning and management and solid state disk.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
265 (Interesting/Helpful), 266 (Too Technical), 267 (Too Basic)

2.) Would you like more articles on the same subject?
268 (Yes), 269 (No)

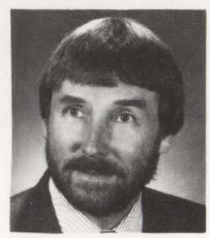
VSE *from page 96*

Summary and Conclusions

VAE provides storage constraint relief for VSE users. At Version 2 the limit is 40MB in three address spaces and at Version 3 the limit is now 120MB in nine address spaces. The architecture provides a shared supervisor, shared SVA and some shared partitions. The shared partitions come at the expense of the private partitions. The implementation uses multiple segment tables to point to a page table. All segment table pointers for shared areas point to the same page table entries. Segment table pointers for private areas point to unique page table entries.

VAE for VSE is a great idea, especially for shops that are not running VM but have outgrown a 16MB single space system. The implementation is relatively trouble-free and operates smoothly. The major disadvantage is lack of communication across address space boundaries; but I'm sure IBM is addressing this issue. You may also need additional real storage but then you didn't expect to get something for nothing, did you?

ABOUT THE AUTHOR



D. Robert Bailey has been active in the data processing world for more than twenty years and has participated in most aspects of the business. He is a senior software developer at Landmark Systems Corp., a leading supplier of measurement and performance software for IBM mainframe systems. Bailey was an independent consultant working for various companies on a diverse range of problems. Recent emphasis has been on VSE but his experience includes considerable work with MVS (aka MVT, VS2) and with personal computers dating back to CP/M days. Landmark Systems Corp., 8000 Towers Crescent Dr., Vienna, VA 22180-2700, (703) 893-9275.

EDITORIAL EVALUATION

Please circle the appropriate numbers on the Reader Service Card.

1.) This article was:
295 (Interesting/Helpful), 296 (Too Technical), 297 (Too Basic)

2.) Would you like more articles on the same subject?
298 (Yes), 299 (No)

CIRCLE #65 on Reader Service Card ▲

VSAM DATA COMPRESSION
Without the CPU Overhead
IAM REDUCES THE SIZE OF YOUR VSAM FILES BY 30 TO 70%

IAM'S FILE STRUCTURE — SAVES 20 TO 40%
IAM uses an advanced file structure which is far superior to VSAM. IAM's supercompressed index, freespace concepts and blocksizes make much more efficient use of disk space.

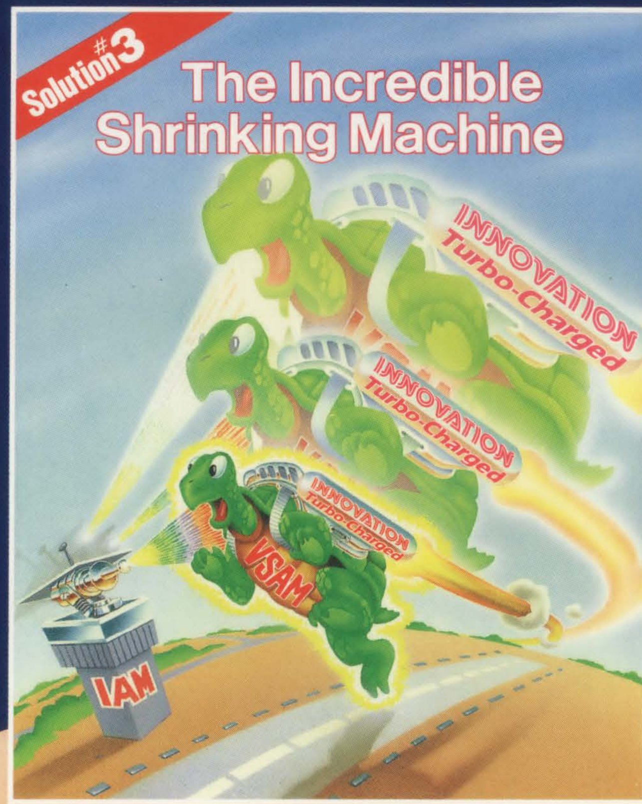
IAM'S DATA COMPRESSION
SAVES AN ADDITIONAL 20 to 50% DASD SPACE
Most files contain records with unused fields or repeating sets of characters. When IAM applies its proprietary compression techniques, the result is an additional 20 to 50% reduction in file size.

IAM's CPU time is dramatically less than competing compression products. In fact, since IAM's CPU time is normally much less than VSAM, IAM with data compression takes less CPU time than normal VSAM processing.

TRANSPARENT
Online systems (CICS), BATCH jobs, TSO, SMP/E and other applications make extensive use of key indexed VSAM (KSDS) files.

IAM is a transparent alternative to VSAM KSDS files, which substantially reduces the impact of VSAM processing in your installation. There are no modifications to programs or JCL to use IAM files in place of VSAM.

AUTOMATIC RELEASE OF UNUSED SPACE
IAM takes the guessing game out of VSAM space allocation. Large amounts of disk space are wasted when users overestimate how much space VSAM requires or how many records a file will contain. VSAM cannot release overallocated space.



FREE VSAM SPACE SAVINGS ANALYSIS*

SPACE SAVINGS

DATA SET NAME	VSAM TRKS ALLOC	VSAM TRKS USED	IAM TRKS STD	IAM TRKS COMPR	% SAVINGS STD	% SAVINGS COMPR	TOTAL RECORDS	AVERAGE LARGEST	MAX LRECL
BIG.CLUSTER	37155	37155	27855	15600	25	58	4754670	233 502	580
CICS.FILE.MASTER. TABLE.CLUSTER	21000	19005	12720	9495	33	50	5068165	150 150	150
NAME.ADDRESS.FILE	9315	8985	6465	1875	28	79	428529	680 2090	2090

The VSAM simulation report displays the current size and data characteristics of your VSAM files and how much space IAM will save you with and without data compression.

**To see your VSAM usage, send for INNOVATION's free VSAM reporting programs.*

Call for a Free No Obligation 90 Day Trial

Supports MVS, MVS/XA and MVS/ESA

Makers of FDR and ABR

INNOVATION DATA PROCESSING
275 Paterson Avenue, Little Falls, NJ 07424 • (201) 890-7300

CIRCLE #97 on Reader Service Card ▲