# Assorted Utilities: Part I

BY SAM GOLOB

In our field, there are several marks of a veteran. The first is a general knowledge and conceptual picture of the different parts of the MVS (or OS/390) operating system. Such knowledge usually takes years to accumulate. Second is the ability (acquired by experience) to quickly isolate the component that is causing a problem and to know where to get help or how to make an adjustment to fix the problem. Third, a veteran should know how to set up the MVS operating system from scratch, starting from the way IBM ships it. Finally, the veteran is constantly gathering an ever-increasing arsenal of tools, always learning how to use them better. You don't measure a veteran's experience in years, but rather in technical savvy.

## SHOPS OF DIFFERENT SIZES AND BUDGETS

Every MVS (or OS/390) installation is different even though they all have the operating system in common. MVS shops are of different sizes and different budgets, depending on what they are trying to accomplish from a business perspective. "Large" shops tend to support huge businesses, and they buy a lot of vendor software and hardware products to satisfy their large needs. Other shops can be very small — either medium-sized businesses or software development shops, with relatively low budgets. Then there are the "medium-sized" shops in-between, supporting large businesses, who buy some vendor software to fit their needs, but they aren't replete with a lot of "fancy systems programmer tools," as the large shops may be.

This makes for differences in how the systems programmer who takes care of that installation's system approaches the job. In a large shop, the systems programming tasks are divided among a considerable number of people. It helps to have a few veterans who can cut through the occasional tough problems quickly, and generally guide the installation's progress in hardware and software. The veterans can also bring the more junior personnel up to snuff, eventually making them veterans also. A small shop is run by possibly one or two people. These people have to be generalists. The medium-sized shop, which is in-between, should always have at least one veteran, and if not, at least one systems programmer who knows how to get access to lots of help.

**You don't measure a veteran's experience in years, but rather in technical savvy.**

Tool-wise, there are many differences between small and large shops. Even with tiered pricing, small and medium-sized shops can only afford a limited number of vendor-supplied tools. Thus, the systems programmers there have to "make do with less." My recommendation for this, which is being followed by more and more people nowadays, is to learn how to use free software tools.

## FREE TOOLS AND UTILITIES

MVS systems programmers are fortunate to have access to many free tools and utilities that are parts of huge software collections such as the CBT MVS Utilities Tape. The CBT Tape, which is independently produced, is distributed by several sources, including NaSPA. Contents of the current version of the CBT Tape are now posted on the Internet at http://members.aol.com/cbttape. A version of the CBT Tape is also part of the NaSPA CD-ROM, which is updated approximately once a year. (**Editor's Note:** CBT Tape Version 416 will be available on NaSPA CD-ROM Volume 1, Version 4.0 due out in second quarter of this year.)

The premise behind the creation of the CBT Tape is that if one systems programmer solves a problem, why should other programmers have to re-invent the wheel to solve the same problem?

This month, I'll discuss one particular collection of assorted tools that was compiled by John V. Hooper. John's tools can be found on File 019 of the CBT Tape. I'm sure that at least one or more of John's dozen or so programs could benefit any MVS shop. This collection was updated on Version 415 of the CBT Tape and is running on an OS/390 Version 1.2 system with JES2 at the appropriate level.

## WHAT'S IN THE COLLECTION?

John Hooper's collection of programs covers the various needs of his medium-sized shop. From a glance at the following descriptions, you'll quickly see that a lot of key areas in the maintenance of an MVS environment are covered.

John has several programs that monitor the blocksize of datasets globally throughout the entire shop. Using these programs, his site has improved I/O efficiency and has helped to eliminate bottlenecks in the company's batch window.

A second group of programs constitutes a general job performance and throughput analysis package. Other programs in the collection perform dataset analysis and monitor DASD volume contents using the appropriate SMF records. Of course, larger shops may accomplish this by buying SAS and Barry Merrill's SMF record analysis package, but John Hooper's shop doesn't have SAS. Therefore, John wrote some assembler programs to perform this task.

John has developed a console automation tool that helps his shop to simplify system startup and shutdown, without using a formal automation package. All of John's programs are all free, are publicly available, and have been running on an OS/390 system. Do they whet your appetite? You can try them too.

There's more. These tools include an operator command to simplify the control of the 3990-3 Cache Subsystem, which normally requires IDCAMS subcommands that are quite complicated. There's a program to report Private Region size below the 16MB line right after IPL time, and to make sure it isn't below a critical threshold. If the Private Region is too small, an alarm in the form of a non-rollable message to the operator console is sounded.

John has a general map of the main storage areas in an MVS system. Although the large performance monitors also have such a map, John's can be run as a batch program to produce a hardcopy report. Another program of John's will track a load module in the linklist or LPA libraries, and display which libraries the program is found in, or where in storage (if LPA) the program is loaded. This program also displays the beginning few bytes of each module it finds, so you can get an idea of which version it is.

Finally, by analyzing LISTCAT ALL outputs for all catalogs, John can trace which DASD volume must contain which datasets. This is useful in an SMS environment where it's not clear on which volume a dataset has been placed, and you can begin to recover when a particular DASD volume goes bad. At least you know what datasets should have been on it.

Some or all of these tools may help you. The following section will provide more detail so you can better judge for yourselves. Please bear in mind that these descriptions are only a beginning; John's utilities have much more depth than I can indicate here.

## THE UTILITIES

**FLSMFDSN and BLKSCAN:** This combination of programs was designed to eliminate bad block sizes of datasets in the entire shop. The FLSMFDSN program analyzes SMF type 14 and 15 records to cover non-VSAM datasets, and type 64 records to cover VSAM datasets. FLSMFDSN has a large collection of sort and selection parameters, making it extremely flexible. FLSMFDSN can be used to flag all the datasets in the entire shop that have inefficient block sizes. After these have been pinpointed, the BLKSCAN program is used to scan all the production and test JCL, showing the origin of the inefficient block sizes. All bad JCL can be corrected, and those jobs will run more efficiently.

**Tool-wise, there are many differences between small and large shops. Even with tiered pricing, small and medium-sized shops can only afford a limited number of vendor-supplied tools.**

**FLSMFJOB:** This program gets its information from SMF type 30 job and step records, reporting on the resources used by a job step or an entire job. A tremendous variety of execution and sort parameters makes this program extremely useful in pinpointing jobs that are hogging CPU or I/O. Once these jobs or job steps have been flagged, they can be made more efficient through small programming changes. Using this program, John's shop has been able to delay a processor upgrade, saving a considerable amount of money.

**COMMAND:** At the time this program was written John's installation didn't have a formal automated operations package, but its operation is large enough (it's a big chain of supermarkets) that system IPL and system shutdown are nevertheless quite complicated. Without the COMMAND program, complete IPL and system shutdown procedures might take up to half an hour apiece. With this program, each of these procedures takes only a few minutes. COMMAND accomplishes this by not only allowing for automatic issuing of MVS and JES2 commands, but by also including DELAY and message REPLY subcommands, so that the proper components can come up (or down) in the proper order, at the proper time, and with console WTOR messages properly replied to. This capability allows any shop to recover from an emergency outage with a minimum of down time, which translates to money saved.

John told me that even with an automated operations package, the COMMAND program is useful for backups of individual applications. Using the COMMAND program, *one* procedure can shut down the application, back up the data, and restart the application automatically. Auto ops packages aren't frequently used for that.

**CHECKPVT:** This program is designed to run immediately after IPL and to report the maximum private area available for started tasks, jobs, and TSO users. After slight system maintenance, it is possible for CSA to be allocated on a different boundary, an entire megabyte lower, without you even knowing it. The private area can be cut by an entire megabyte of virtual storage, making some large jobs or CICS regions abend when they are started later in the day. This is not particularly pleasant and can necessitate an IPL for the backout of maintenance in the middle of the day. CHECKPVT will check the private region size at IPL time and will immediately write a non-rollable message to the operator's console if it is less than a pre-determined threshold value. The production disturbance will be nipped in the bud.

**FLSMFCAT:** This program gets information from SMF records to show ICF catalog activity at the dataset level. Sometimes, when a dataset is created or deleted, and the application fails, it is not easy to tell who created or deleted the dataset. The SMF ICF catalog information is the best way to tell what happened.

**FLSMFSRT:** This program produces reports from SyncSort SMF records. This program allows John's installation to monitor the larger sorts in production and test jobs. Of course, the program is only useful if your shop uses SyncSort as its sorting utility product.

**FLVOLLST:** This program is designed to print a report listing all of the datasets on a volume based upon information from the system catalogs. This list could be critical in case of a DASD failure that destroys the VTOC on the volume. With volume pooling now available through the use of DFSMS and other program products, it is not always easy to determine the datasets that are on a specific volume.

The input to this program must be the output from an IDCAMS LISTCAT command. It is expected that an IDCAMS 'LISTCAT VOL CAT(user.catalog.name)' command will be executed for each catalog in the system. The contents of these reports can then be passed to this utility program to produce the report by volume and dataset name.

**MODLOOK:** This program is designed to run as a TSO command, started task, or a batch job to look up the selected module(s) in the system linklist or link pack area. If the module is in the linklist, the linklist library name will be displayed. If the module is in the link pack area, its address will be displayed along with the name of the resident area in which it is located such as PLPA, FLPA, ECSA, etc. The first part of each module is displayed also, since it can contain date, time, or copyright information that may be of interest.

With many libraries now in the system linklist, it may not always be apparent which dataset contains what program. More importantly, it may be difficult to determine which library contains a module if duplicate module names exist.

**SMAP:** This program is designed to print the starting address, ending address, and size of each of the main storage areas in the MVS system. This information can be displayed if your site has a monitor. Also, this program can run as a batch job producing a hardcopy report.

**JES$LF:** The JES$LF program is a JES2 Exit 5 module designed to process the $LF command when entered. It will give detailed information at the output group level for jobs awaiting print. This is, in effect, a detailed version of the $DF command.

In summary, John is another example of a practitioner who has developed several inventive tools to help his shop and is gracious enough to make the tools generally available to the public on the CBT Tape. I hope this information will encourage you to look at these utilities in more detail. Good luck. I'll have more of these tools to show you next month. ts

**NaSPA member Sam Golob is a senior systems programmer. Sam can be reached at sbgolob@aol.com or sbgolob@ibm.net.**