When IBM Says You Can't ... Sometimes You Can

BY SAM GOLOB

Everyone in the computing field must live with limitations. For example, application programmers feel hemmed in because their authority to access the installation's data is constantly limited by the security system. We systems programmers also experience that feeling to a lesser extent. But such a limitation, however irritating, is not necessarily bad. It stops people from destroying or altering mission-critical data that they shouldn't have access to.

In contrast, here's a different kind of limitation, which in my opinion, is a very bad thing. Sometimes you wish you had a utility to perform a certain function but you're told that such a utility doesn't exist. My answer is that it's not necessarily true. In fact, it might be amazingly easy to write an assembler program or a REXX exec to do exactly what you want to do. Furthermore, someone else might have already done the work. When you experience this frustration, you can often do something about it, and it's legitimate for a systems programmer to refuse to permanently accept that kind of limitation.

In truth, many IBM "no-no's" are actually doable. Often, if you sit down to do the research concerning the internal structures that you want to manipulate, you'll see that there's nothing to prevent you from writing a program to do the job.

I've surprised myself by doing some fairly clever things that I never thought I could do. My SYS1.BRODCAST manipulation package (CBT Tape File 247) is an example. IBM says you can't display and clear other people's messages from the SYS1.BRODCAST dataset. However, I've written a bunch of utilities to do it easily and safely. Of course, this took time. It follows, then, that our problems could be solved if we could all divide the work and share the results.

Well guess what? There are large collections of free utilities already available that are being added to continually. One of the biggest single collections is the CBT MVS Utilities Tape, a huge collection that's independently produced and is available from the NaSPA office as well as several other sources. Once you have a CBT Tape of your own, you can make copies for all your friends using the COPYMODS program on File 229. There are other similar free collections. File 071 of the CBT Tape contains the documentation files for a large number of them.

Many IBM "no-no's" are actually doable. Often, if you sit down to do the research concerning the internal structures that you want to manipulate, you'll see that there's nothing to prevent you from writing an assembler program to do the job.

Let's look at another specific example. IBM says you can't add directory blocks to a partitioned dataset without reallocating the dataset (with more directory blocks) and copying all the members from the old partitioned dataset to the new one. There is a RACF drawback to having to allocate a new pds and delete the old one. You need ALTER access to the dataset name. Upon studying the situation, you can see that it doesn't have to be that way. If you were able to move some members from the beginning of the pds to the end, and reformat that space from the beginning into new directory blocks, you wouldn't have to reallocate a new dataset; you could reuse the old one, and you'd only need UPDATE access, not ALTER. This could save a lot of trouble.

Well, there exists a free utility that does this. You point the utility to the pds and tell it how many more directory blocks you want, and it tells you which members have to be moved to the end and asks you if you want to continue. When you say "yes," the members get moved, and the directory gets extended. It's that simple, and the process works very reliably. As I mentioned, the utility only requires UPDATE security access to the dataset; you don't have an enqueue problem deleting the dataset and this utility does not have to run APF authorized.

Where is this magic bullet? It's actually something I've mentioned many times in this column. It's the PDS program package from File 182 of the CBT MVS Utilities Tape. The PDS package (and its vendor-supported successor STARTOOL from SERENA International) actually performs many more functions that IBM says are impossible.

Once the PDS program is set up (it runs as a TSO command), you point it to the dataset you want to change. Next, you enter the subcommand: FIXPDS EXPANDDIR(nn), where nn is the number of directory blocks you want to add. The PDS command then comes back with a prompt, listing which members would have to be moved, and asks if you want to continue. If you reply "Y" or "YES" to the prompt, PDS moves the members, reformats the additional directory blocks as you requested, and reports the new number of free directory blocks that the pds now has. So much for IBM's limitation.

ADDING A NEW DATASET EXTENT AND OTHER JOBS

I mentioned that the "PDS" package will do other things that IBM says you can't do. One of them is to add a new extent to a dataset, no matter what the FORMAT 1 DSCB (i.e.,VTOC entry) says, about how much the secondary space of the dataset is set to. Even if the secondary space of the dataset is marked zero tracks, you can add a new extent as big as you want. You simply point the PDS command to the dataset as mentioned previously and enter the subcommand:

```
FIXPDS ADDTRK(nn) or FIXPDS ADDCYL(mm)
```

PDS will do as it was told and add a new dataset extent of nn tracks or mm cylinders.

PDS doesn't have to run authorized to do any of this. It does some other tricks, most of which you could probably also do if you "misuse" JCL. Suppose you've pointed the PDS command to a dataset. You can enter FIXPDS RECFM(xx) or FIXPDS LRECL(rr) or FIXPDS BLKSIZE(bbbb), and the PDS command will listen to you after you've answered "YES" to its prompt. These alterations to a dataset sometimes have to be done under some special circumstances, and I'll show you one.

This modern example involves FTP. I've discovered that I can FTP to an MVS computer site that has an Internet connection from my home PC. I can often get to MVS by telling the PC FTP program my TSO userid and password. Suppose now that someone has FTP'ed a downloaded pds to me that's in TSO XMIT format, which has to be LRECL 80, and I want to upload that pds to this MVS system. I simply FTP from my PC (through the PC's FTP program) to the MVS TSO account and upload the file.

Unfortunately, it's not that simple. In order for the TSO RECEIVE command (the opposite of TRANSMIT or XMIT) to reconvert the file into a pds that's the image of the original pds from the sending system, the file has to be LRECL 80. RECEIVE only "understands" an XMIT file with LRECL 80. However, my PC FTP program only sends the file to TSO in blocks of 4160 bytes, and the resulting file is marked on MVS as having LRECL 4160 and BLKSIZE 4160 — 4160 is a multiple of 80 — and actually, all the data is intact on MVS. Only the "official LRECL" or "marked record length" of the data is wrong. As I said, RECEIVE can't restore this file as it's marked. However, I can just point the PDS command at the file and say:

FIXPDS LRECL(80). Voila, the data is marked LRECL 80, BLKSIZE 4160 and it's usable by the RECEIVE command. Quick and straightforward. Wow!

If you could write a program to create a new copy of the JES2 exit program in the proper environment (the JES2 address space, or the CSA or LPA areas of common storage) and re-point the LMT entry to the new copy, you've basically solved the problem, minus a few details.

MORE ON CHANGING DATASET ATTRIBUTES

There's another free utility that can change dataset attributes. This one is called CDSCB (or Change the DSCB), and it's a TSO command that must run authorized. CDSCB can be found on the CBT Tape on File 300. CDSCB has to run authorized because it zaps VTOC entries directly, as opposed to the PDS command, which works by writing a dummy addition to a dataset while forcing DCB attributes to the dataset, like you can do using JCL.

CDSCB was written by Bill Godfrey, with the idea that if you're changing a dataset's attributes by zapping the VTOC, you want to be safe and not change the wrong VTOC fields. CDSCB makes the process more foolproof. CDSCB can also be run under TSO-in-BATCH. You can get the APF authorization in batch by running CDSCB from an authorized library and mentioning that library as a single STEPLIB dataset.

I've used CDSCB to add larger amounts of secondary space to a whole volume full of IBM datasets. As most of you probably know, IBM's shipped dataset allocations contain barely enough space to hold their shipped data. If you have a lot of maintenance to APPLY or ACCEPT, you can get many annoying utility failures due to inadequate secondary space and inadequate allocation of directory blocks in the IBM datasets on the DLIB and TARGET packs.

By using CDSCB and the PDS command under TSO-in-BATCH, I can get a mass SMP/E APPLY or ACCEPT job to run, even though a lot of material has to be added to the target libraries or the DLIBs. Here's how. Obtain a list of all the datasets on the DLIB or TARGET pack. Then edit the list of names to say the syntax: CDSCB 'SYS1.whatever' SPACE(30) ALLOC(TR) VOL(volser) for each dataset name. This will force the "official" secondary space allocation to be 30 tracks for all the datasets. If the dataset happens to be large or needs additional space, you can edit the settings in CDSCB to make the potential space allocation larger. If you want to force secondary allocation in cylinders, just substitute SPACE(2) ALLOC(CY) for SPACE(30) ALLOC(TR). If CDSCB ran successfully, it will display the message: "CHANGED" after each invocation. Otherwise, it will say "NOTHING CHANGED."

To add more directory blocks to all the datasets on the pack, you can run the PDS command under TSO-in-BATCH. The proper commands will take up at least two lines per dataset. The first line will state: PDS 'SYS1.whatever'. The second line could state: FIXPDS EXPANDDIR(30) to add 30 directory blocks to each of the datasets. Under TSO-in-BATCH, PDS answers the "YES" prompt automatically. You could add a third line for some of the datasets that don't have enough space:FIXPDS ADDTRK(60). This would physically create a new extent of 60 tracks for that dataset, and not just mark the potential size of a future new extent, as the CDSCB "SPACE" keyword does.

At the end of this process, you should have a whole pack full of adequately allocated system datasets, and your mass APPLY and mass ACCEPT troubles will be solved or at least be more easily handled. Of course, make sure that there is enough actual free space on the whole volume, so your system datasets have room to expand if necessary during the SMP/E run.

A JES2 EXIT LOADER

Anybody who writes JES2 exits knows that IBM says you have to recycle JES2, or IPL, in order to change *which version* of a particular JES2 exit you are running. IBM does supply a system command to ENABLE or DISABLE a particular JES2 exit. However, if you DISABLE and then re-ENABLE one particular exit, you always get the same

version. You can't plug in a new version of a particular JES2 exit unless you bring JES2 down and up again or re-IPL. At least IBM says you can't!

Who has a problem with this? The systems programmer who is testing a new JES2 exit or is modifying one. To try out a new fix, you have to disturb everyone on that particular system by recycling JES2 or IPLing. Even if you're testing your exit on a test system, there may be many other programmers testing their stuff at the same time, and you'll be disturbing their work.

If you know JES2 internals to some extent, the situation is not at all insurmountable. The reason why you can't change versions of an exit is that JES2 points to its load modules (including the modules belonging to the various exits) with an address table called the Load Module Table (LMT). The LMT contains address entries that point to the entry points of all the JES2 modules. IBM simply doesn't provide a facility to re-point an LMT entry to a different module without a restart of JES2.

If you could write a program to create a new copy of the JES2 exit program in the proper environment (the JES2 address space, or the CSA or LPA areas of common storage) and re-point the LMT entry to the new copy, you've basically solved the problem, minus a few details. You could code your program as a JES2 Exit 5 routine (JES2 command modification) so it could be invoked from the console as a new JES2 command.

IBM has refused all suggestions to release a dynamic JES2 exit loader program. I'm not sure why. One guess would be that IBM doesn't want to take the responsibility if someone irresponsibly disturbs production processing by altering the version of an exit that's running. That doesn't seem to be so likely, because even today, an operator can disable any JES2 exit with a command. In any case, IBM has left the poor systems programmers who code JES2 exits high and dry.

Not anymore. There's a partially effective JES2 exit loader program on File 196 of the CBT Tape. This exit loader will load a new version of any JES2 exit that runs in the JES2 address space environment (which includes many of the exit points). Until now, the public didn't have an exit loader available that would work for exits in all JES2 environments — CSA and LPA included. Now, an expert JES2 programmer, Bob Break, has written a fully functional JES2 exit loader program that's on the JES2 SHARE Tape (from Jack Schudel of the University of Florida, schudel@ufl.edu). Bob Break's program will be on File 198 of Version 418 of the CBT Tape. You can also get a copy of the new exit loader at http://members.aol.com/cbttape/jes2xldr.xmi. Our problems are solved!

Before closing, I'd like to add that there are many more things that IBM says you can't do but which you can. If you have such a question, look at the CBT Tape documentation on the web, or subscribe to the IBM-MAIN newsgroup and ask your question (see my April 1998 column). With persistence, you'll find someone who can point you in the right direction. Good luck. See you next month.

Documentation about the CBT MVS Tapes can be found at http://members.aol.com/-cbttape. ts

---

Sam Golob is a senior systems programmer. He can be contacted at sbgolob@aol.com or sbgolob@ibm.net. Sam also participates in library tours and book signings with his wife, author Courtney Taylor.