Tuning Your SDL

BY LEO J. LANGEVIN

Let's say you install the latest release of VSE/ESA. You also apply the latest tapes containing vendor product changes. You spend a lot of time fine-tuning the IPL procedure, and everything goes well during the testing process. Finally, the day comes when you cut the new release over to production. However, by lunch time, CICS comes to a halt because you've exhausted all of your 24-bit GETVIS. Oh sure, many of the programs and even CICS use 31-bit storage, but you still have a lot of older code around and some vendor packages that still use the "below-the-line" area.

> Finally, the day comes when you cut the new release over to production. However, by lunch time, CICS comes to a halt because you've exhausted all of your 24-bit GETVIS.

You then use the MAP command on both the old and new systems only to discover that you've lost an entire megabyte of 24-bit storage on the new system! The reason is simple: All partitions will always begin on a 1MB boundary. You also see that the beginning of SVA-24 has not moved, or has moved only a little bit, so the real culprit might be the growth of the modules that reside in the 24-bit SVA. Let's examine this a bit closer.

The partitions sit on top of the supervisor, the 24-bit SVA, as well as some other areas. Let's say that the total of those 24-bit areas only needed 5.9MB of storage, so there was a .1MB buffer between the end of the lower 24-bit area and the beginning of the partition. Now you install a new release of VSE and some of the SVA modules grow, or perhaps the supervisor gets bigger, or perhaps some vendor modules increase, adding a .2MB requirement. Now you end up with 6.1MB of 24-bit reserved storage, and since there is a 1MB boundary requirement, the other .9MB remains unavailable, causing the lower 24-bit area to use 7MB instead of 6MB.

So what can you do? First, you can tune the "SET SDL" portion of your IPL procedure. Make sure that the same module name is not being loaded more than once. Some vendors will provide canned procedures that you need to include as part of your start up. Some of these procedures will include the same names that you already use. While this will not produce a duplicate system directory list (SDL) entry, it will allocate GETVIS and store a new copy of the program in storage and update the pointer in the SDL. Be careful — I know of one shop that was loading $JOBACCT eight times!

Next, produce a 24-bit SVA spreadsheet. Just produce a "LISTDIR SDL" report from LIBR and import it into your favorite spreadsheet program, such as MS-EXCEL. Then, sort all of the entries by location address. Delete any entries that have a 31-bit load address as well as any headings and other non-important lines. If you believe that you only need to do minor tweaking to reduce the size of the SDL (the directory of all entries that are in the SVA or fast library pointers), check the number of entries that are not in the SVA (those which simply have an SDL pointer). This is used primarily to speed up processing. But are all of those entries needed? For example, if your shop doesn't use ICCF, do you really need all of those ICCF pointers in the SDL if you are using them less than once per minute?

Next, you need to calculate a new spreadsheet column that will tell you the ending address based on the hexadecimal starting point and the length. Check to see where the dead spots are. There's bound to be a few bytes between each one due to program boundary alignment. However, if there are large gaps, it may be because you are loading the same program multiple times or a vendor program is pointer swapping.

Figure 1: $$A$SVA

```
$$A$SVACSECT
          DC    CL8'$SVACSC'           SYSTEM PHASES
          DC    CL8'$SVAICCF'          ICCF PROGRAMS
          DC    CL8'$SVALOG'           ACCESS CONTROL LOGGING
          DC    CL8'$SVASEC'           SECURITY
          DC    CL8'$SVABAM'           OTHER ACCESS METHOD MODULES
          DC    CL8'$SVAVSAM'          VSAM MODULES
          DC    CL8'$SVARCF'           CICS/RCF
          DC    CL8'$SVA0000'          DUMMY ENTRY
          DC    CL2'/*'                END-OF-TABLE INDICATOR
          END
```

Figure 2: $SVAICCF

```
SVALLIST $SVAICCF,                                           X
          (DTSISEOT),                                        X
          (DTSIPWR),                                         X
          (DTSPROCS),                                        X
          (DTSSBMT1),                                        X
          (DTSCDUMP),                                        X
          (DTSXTRCT),                                        X
          (LINKNGO)
END
```

And now for the dangerous part: Make sure you can recover your IJSYSRS.SYSLIB from a standalone environment in case you mess this one up!

During the IPL, VSE will look for $$A$SVA. (It's actually a longer process than I have space or time for in a single column, so I've included some short cuts in these explanations.) Figure 1 provides the source code for that module.

Each of those entries is the name of a PHASE that resides in IJSYSRS.SYSLIB. If you do a LIBR "LIST phasename" of $SVAICCF.PHASE you will see that DTSISEOT, DTSPROCS, DTSSBMT1, DTSCDUMP, DTSXTRCT, and LINKNGO will automatically be loaded into the SVA. If you comment the line for $SVAICCF and reassemble $$A$SVA, then those entries will no longer be loaded.

However, ICCF requires DTSCDUMP to be in the SVA. So how do you get around this? Well you can either manually add "DTSCDUMP,SVA" to your standard "SET SDL" startup procedure or you can modify the $SVAICCF module. The source for $SVAICCF is shown in Figure 2. You can modify it to a single entry for DTSCDUMP and save 24-bit SVA storage by leaving out the other entries. If you aren't going to use CICS/RCF, then you may want to comment out $SVARCF from $$A$SVA. If you have SEC=NO as part of your system definition, then you may want to comment out $SVASEC from $$A$SVA.

SOME FINAL TIPS

1. It's a good idea to list the contents of $SVACSC, $SVAVSAM, etc. Compare these entry names to what you are manually having loaded as part of a "SET SDL" process. Be sure that you are not performing duplicate loads. For example, $JOBACCT is in $SVACSC and some people will still keep reloading this entry again and again.

2. Understand what the function is of each module. If it's only used for VSAM managed SAM, then you may be able to remove it from $SVAVSAM.

3. IBM does not provide any recovery if you remove an entry from automatically being loaded into the SVA and the IPL process requires it. The IPL will stop right after a fatal error message appears, indicating the name of the missing phase. You will need to restore your IJSYSRS.SYSLIB. An easier way to do this if you have multiple VSEs that share the same DASD is to have the problem IJSYSRS.SYSLIB identified to an alternate VSE system as IJSYSR1.SYSLIB. Then you can copy the original phase and just re-IPL. Now keep in mind, it's not just IBM that has this recovery limitation. Find out from your vendors which modules are desired to be in the SVA and which ones are required to be in the SVA.

NaSPA member Leo J. Langevin is the senior developer of NFS for VSE from Connectivity Systems. He's worked with VSE since its inception. He can be reached at leo@tcpip4vse.com.