Expanding Your Horizons

BY SAM GOLOB

I think everyone in our business of systems programming will agree on one fact: In order to do this job well, you have to know a lot! How do you get to know a lot? It doesn't just happen. To obtain the necessary knowledge, one has to make a concerted effort over a long period of time. This month, I'll examine how to increase the efficiency of making this effort to learn more, how to motivate yourself to continue, and how to have fun at the same time.

We systems programmers, like almost everyone else, learn things on the job by being work-driven rather than by being self-driven. Of course, it's good to learn everything you can from the tasks you've been assigned to do at work. However, if this knowledge is supplemented by some of your own initiatives, it gives you a real edge. You can get "good" much quicker. It's also more fun when you set your mind to learn something and then do it.

I'm very fortunate that at the beginning of my systems programming career I had a teacher named Jeff Broido whose main outstanding characteristic is that he is a very unselfish and considerate man, besides being excellent technically. For three years, even when we were working at different places, I'd call Jeff every day and tell him what I did on the job. Even when I felt I knew what I was doing, Jeff would tell me the history of the component or provide some background about the subject that I hadn't known. And, if I didn't know what to do, Jeff would, of course, enlighten me as far as he could. Having Jeff's influence early in my own career inspired me to help other people also as much as I could. The fact that I write this column comes directly from Jeff's inspiration and consideration. If you find this column useful, you all "owe him one."

We systems programmers, like almost everyone else, learn things on the job by being work-driven rather than by being self-driven.

Jeff had marvelous "rules" for a systems programmer to live by and I'd like to share some of them with you. His first and foremost rule was that you should spend time each day learning something new. Whether it was five minutes or a half-hour, the fact is we're not kept "103 percent busy" every day; there's usually some "lull time" on many days that can be spent absorbing some fact or learning a new technique. For example, Jeff studied the ISPF tutorial panels until he knew all of them. Then, when he had a tricky editing job to do, he could do it quickly instead of taking all day. This gave him even more time to learn more new things.

The result of consistently applying Jeff's first rule is this: Instead of being "103 percent busy" all day, you get to be about "50 percent busy" most of the time. Call it "increased CPU capacity," but whatever it's called, it sure makes life easier. After my first two or three years, I've spent most of my career being "50 percent busy" as a result of this training. All of my jobs were completed on time and without errors. It pays to become knowledgeable, but it also pays to keep learning new things.

The thinking behind "Jeff's first rule" is to replace a pattern with another better pattern. What's the first pattern that everyone follows? "Learn just enough to do the job that's expected." So what do we all do? (And I'm included in this.) We get all panicky trying to learn just enough to get the job done and then we relax and "settle in," which consolidates our newly learned skills. Then, when there's another job that requires us to learn something new, we get panicky again. Then we settle in again. It's a cycle.

Jeff's first rule serves to break this cycle during the "settling in" part — not letting it last too long. The good part of the "settling in" process is that it strengthens our existing knowledge. However, the bad part is that if it lasts too long, we're missing a golden opportunity to learn something new. If we force ourselves, by creating the time to learn something completely new every day, we become self-driven to expand our horizons, rather than remaining purely work-driven.

READING MANUALS

I have to admit that the process of reading manuals has changed since the days when I learned this principle from Jeff. In those days, all of the manuals were printed on paper. Today, many of them are on CD-ROM and the information has to be read from a terminal screen. In spite of these differences, however, many of Jeff's ideas still remain largely applicable today. Jeff taught me how to absorb information from an entire manual relatively quickly.

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

The process seems easy and innocent enough, yet there's a vast difference between "knowing about it" and actually doing it. In order to appreciate how really good this idea is you have to actually perform this task several times.

When you need to learn things from a certain manual, here's what you do. Start at the beginning of the manual and start turning the pages, one at a time, spending about a second or two looking at each page until you go all through the entire manual from the beginning to the end. For a 300-page manual, this should take about an hour. Only afterward do you start reading the things you're actually interested in, in detail. The difference that the "initial hour" makes is astounding. A quick pre-knowledge of the material supplies a perspective. This includes a picture of how each topic relates to other topics, a glance at the technical terms and names used by the software, and a general overview. Presumably you've also spent a short time looking at all the pictures.

After you've gotten into the habit of doing this all the time, you'll find that you can completely absorb new topics amazingly quickly. You'll surprise your coworkers with how much you know, without spending significantly more time reading than anyone else.

USE TSO

I guess that this idea was more novel in the early days of MVS than it is today, but it's surely just as applicable nowadays, maybe even more so.

In the earlier days of the operating system, batch jobs were the way of doing most everything. To get the computer to do your bidding, you'd submit a job. At one time, there was no choice. Once TSO was invented though, its facilities eventually grew to the point where 90 percent of what we accomplished could be done directly through our TSO terminal sessions, rather than by other means.

TSO has several major advantages over batch jobs. For one thing, you don't have to grab a new initiator every time you want to do some task. Your TSO session has its own address space and keeps holding on to it. Secondly, you don't have to wait for printing services. The result comes back right to your own screen, generally very quickly. Thirdly, TSO greatly reduces the time it takes to perform short tasks and you don't have to wait for the results. It's true that for long tasks batch operations are usually better and they don't tie up your terminal. But Jeff's point in this area is that if you expand your collection of TSO-based tools and your skill with them you'll soon be overjoyed at how quickly you can get your work done and how little batch work you'll actually need.

The latest version of the free PDS package can be downloaded from the web at http://members.aol.com/freepds.

Spending time each day learning such tools will pay you back with much more available time, so you can learn even more. That can make a difference between being just good and being a real "crackerjack" at this trade.

In the late '70s, when handy TSO-based tools were not so generally available, Jeff taught himself how to write his own TSO command processors, which are what programs running under TSO are called. By writing many assembler-based TSO programs to do his tasks, Jeff was able to make his working life far more efficient. Nowadays, people would probably do most of this work in REXX, which of course, wasn't available then, although CLISTs were. When one of us "ordinary guys" would watch Jeff work, it would blow our minds. He had so many inventions and he was so fast!

Some of Jeff's programs can be found on File 423 of the CBT MVS Tape, a huge collection of public MVS systems programmer software that's independently produced. The NaSPA CD-ROM, which comes out annually, contains version 416 of the CBT Tape. For up-to-the minute information, visit my web site for information. I'm currently trying to find a web-based home for the intermediate updates to the tape. If you're a member of SHARE, you can go to the members-only section of www.share.org.

Nowadays, many of the tools that Jeff had to invent are already available on the CBT Tape, in vendor-supported products, or they can be downloaded from sites on the web. There is a galaxy of TSO-based tools to help you work more efficiently. For example, if you learn to use the TSO-based UCLA Fullscreen ZAP program (File 134 of the CBT Tape) instead of IBM's AMASPZAP (superzap) batch program, you'll see what you're doing much better and you'll probably save a lot of time getting the work done, too. If you install and learn the TSO-based free PDS program package on File 182 of the CBT Tape and now on the web at http://members.aol.com/freepds), you'll be able to do many of your tasks more quickly. (If your shop is licensed for the vendor product called STARTOOL from SERENA International in Burlingame, Calif., which is an outgrowth of free PDS, you just have to learn STARTOOL.) If you've installed the single load module (from File 183 of the CBT Tape) called SHOWMVS in your TSO/ISPF session, you'll be able to learn more about your system in five minutes than EDP auditors used to be able to find out in several hours of digging. The "unofficial" IBM program called TASID, written by IBM'er Doug Nadel, which you can download from the web, will also supply much of this information.

Spending time each day learning such tools will pay you back with much more available time, so you can learn even more. That can make a difference between being just good and being a real "crackerjack" at this trade. The time could be spent installing a new tool from the CBT Tape or elsewhere. It could be spent reading a manual (in Jeff's "efficient" way). If you do something, anything, to expand your horizons, you've done a very positive thing — you've helped yourself.

Good luck. See you next month. ts

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