MANAGERS, YOU CAN CONTROL RESPONSE TIME!

(It's a management function, and nobody else can do it for you.)

A presentation for the HP General Systems Users Group at the North American Meeting, February 1979, in San Jose.

By: John Beckett
Director of Computer Services
Southern Missionary College
Collegedale, TN 37315 (615) 396-4303

Abstract:
In this paper are presented a number of concepts and techniques useful to the non-technical manager responsible for operation of an HP 3000, in providing service of a predictably acceptable nature. The primary strategy stressed are optimizing applications and properly distributing workload on the computer. SLS, a program which helps prepare job streams, is presented as a primary means of offloading the CS queue by assuring that jobs will run properly through extensive variable validation.

Southern Missionary College acquired an HP 3000 series II model 9 in March, 1977. Our previous experience was with the IBM 1130 and the HP 2000F. It took us (with the help of a few dedicated students) about three days to run into response time problems. We discovered several things very quickly:

1. It was almost impossible to determine how much of the time our machine spent swapping, and who was requiring the resources.

2. HP thought more memory would solve the problem. Since we already had 320k and 512k was the maximum for the machine, we questioned that.

3. About eight sessions was the maximum we could handle without experiencing severe response delays--in exess of 10 seconds between lines in EDITOR.

We found ourselves surprised and confused. We had spent quite a bit of money on the HP 3000, and it looked like the machine would not do the job we needed.

This presentation is what we learned along the path towards the current situation - in which users experience predictable response time without our mandating restraints they consider unacceptable.

To appreciate that accomplishment, you must realize that we have over 400 users on our machine - the majority of them writing their own programs. We have almost nothing in the way of jobs run by our operations staff (clients submit everything if they have terminals). We have no fixed job schedule, so Accounting can run a six-month general ledger any time. Sessions normally run 25-30.

Why is response time a management problem?

Because it affects the success of your mission:

* Direct payroll costs to employees while they wait.

* Damage to the service of your organization, through clients waiting on response time (try reservations with an airline).

* Distractions while people wait - past one or two seconds, they start to think about something else.

* Frustration factor resulting directly from slow response.

* Scheduling disruptions because your computer is overloaded.

*** You can't afford response time problems! ***

In any management problem, there is a classic cycle we go through:

1. Identify the problem, as near the root as is within our jurisdiction. (If > first pass, pull previous analyses from file.)
2. Determine what we can do about it.
3. Do the thing identified in #2 above.
4. Analyze what we have accomplished.
5. File the analysis where we can find it later.
6. Go to 1 (A manager's work is never done.)

That cycle is one dimension of the solution to any problem. The other is the specific areas of inquiry and techniques you may use. What spend our problem-fixing time on should cover both of these axes well.

For the remainder of my presentation, I'd like to go over topics we have found fruitful. Please feel free to interrupt me for questions any time, but I'll leave time at the end for questions and discussion also. That would be a good time to bring up a success story from your shop, or a technique you have found useful.

It seems that the first reaction anybody has to a slow system is that you need to buy more memory. We found that this is by no means a foregone conclusion. You can use more memory ONLY if your problem is excessive swapping. In practice, we found that memory sizes below 512kb are more likely to benefit from adding of memory than sizes above. But there are a lot of "givens" in that rule of thumb, so don't take it as necessarily applying to your case. The only authority I am presently prepared to believe on the issue of the need for more memory is IOSTAT2. I would set it up for an update cycle of one or two seconds, and see what percentage of the time disc accesses are for MAM compared to the total. If this is over 50%, more memory will help noticeably. If it is 25% or less, I have serious doubts if ANY memory upgrade will of itself make a noticeable impact on session response (although it might help with batch thruput).

We upgraded our 320kb machine to 448kb in April, and subsequently obtained a Series III upgrade to 768kb in August (both 1979). In both cases our users noticed substantial improvements in service. Subsequent investigations indicated that 98% of the time, more memory would not have an appreciable impact on response time. So for the present we have no plans to increase memory size.

I would like to comment briefly on the upgrade to a Series III. The upgrade product is one of which HP is justifiably proud. It is reasonably-priced, and went in at our shop with less downtime than a regular preventive maintenence session. The CE spent the majority of his time getting the nameplate changed, as the CPU part was very little work at all. Performance is far better, due to our need for memory being fulfilled. MEMLOG insists we haven't dropped a bit in three months, where we were losing twenty per day before. That helps crashes, which in turn helps you get the work out.
But that is only one aspect of the task of improving and maintaining good response time. Memory is one limiting factor. It is likely that you are being limited far more by other factors that are less expensive to attack:

Tools and Techniques for Managing the HP 3000

1. Monitor and Adjust System Tuning Parameters.

By this I don't mean changing QUANTUM, TPRI, DPRI, etc. Some will swear by a particular magic formula of these. Our approach has been to use what HP recommends because that's what MAM plans on. Also, we run exactly one batch job (other than terminal-servicing jobs). This reduces slightly our thruput when sessions are down, but that is exactly when nobody is complaining anyhow.

The following programs and commands are useful in this context:

A. TUNER2 (library) Shows status of various tables. This helps you know if you have a bottleneck somewhere inside internal tables in MPE. If TUNER2 asterisks a table, increase its size (with the possible exception of JPCNT and SBUF).

B. SOO (library) Shows where your CPU time is going. Sorry, it does not show time going to MPE, Spooling, and waiting. Best we have there is a calibrated eye focused on the CIR.

C. SHOWVM (library) Shows who is getting the memory in your CPU.

D. IOSTAT2 (library) Shows where your disc accesses (probably the source of any problems you have) are going. Paticularly useful for determining how much swapping is going on, and how well your files are distributed.

E. REPORTER (library) Formats a REPORT @.@ listing to show percentages. Saves lots of time with a calculator. We have a version called REPFMT which allows you to set up "super-accounts", which are groups of accounts you want to group together in categories.

Numbers you should keep track of:

1. CPU seconds your system logs per business cycle, and where the demands fall within that cycle.
2. Same thing, only for disc space.
3. Proportion of both which go to each major service category.

That old manager's standby--the graph--will tell you much about what is going on in your system, and enable you to spot trends and deal with them before they become critical. Time and space are both important indices of the resource your service represents. Being understandable, they will be an important factors in your useage reporting (or charging as the case may be) system.

F. MEMLOGAN (PUB.SYS - supported) Tells you if you are having memory problems.

G. SHOWQ (console command) Tells you the state various processes are in. We haven't seen any understandable guide to its use yet. For every rule of thumb, we've seen situations in which the rule leads one astray.

H. SYSDUMP (PUB.SYS - supported) The only "good" way to change system parameters. Make a tape with null (CR only) date with the changes, then COLDstart from that tape.

What most of these tools do is tell you how things are going. We use all of them extensively. REPORTER, for instance, is run daily so we can plot CPU seconds and disc space useage curves.

2. Optimize Applications.

In my opinion the greatest untapped resource in the HP 3000 universe is application optimization. If your analysts and programmers haven't studied the topic (they don't need to be able to write SPL), get the following materials and have a study session. You will be repaid many times in performance. I strongly recommend:

A. "Applications Design and Optimization for the HP3000", a document prepared for internal use by HP's Software Engineers. It is written in a style your programmers can understand. Get a copy, whatever the cost.
B. "HP 3000/Optimizing On-line Programs" by Robert Green, in the HP GSUG Proceedings/Denver 1978. Mentions some additional techniques that are a help.
C. "System Performance Measurement and Optimization", by Jim Squires and Ed Splinter, in the HP GSUG Proceedings/ Denver 1978. Rather technical, but was very helpful to us in understanding what our system was doing.

3. Distribute your workload properly.

It is quite probable that much of what your CPU does is at a priority far above what it needs to meet user requirements. Your time will be well-spent researching where this resource is and exploiting it. We have a rule that if output from a job is completed after the client goes home from work, that job goes into the overnight queue, period. It is cheap to maintain free disc space to spool job output. With this resource, you can use night hours (without an operator!) to do lots of things that would destroy system response if run in the daytime.

Specific areas to watch:

A. EDITOR and COBOL. Don't use the COBOL compiler at all in the CS-queue. Use EDITOR only while the system is lightly loaded. Use KEDIT (see swap tape) to make minor changes, as you can thereby avoid Text/Keep overhead. QEDIT is an even better answer to text editing overhead. We use it almost exclusively.

B. Interactive use of QUERY, especially serial finds and reports. Such things are relegated to batch access in our system.

In the case of programs that must for some reason be run interactively but "crunch" a lot, switch the priority dynamically. Use CS priority for interaction with the user, DS for crunching.

4. Watch for and correct recurring failures.

How many times have you wasted computer time because of a crash or bad input data? We found at one point that crashes alone were keeping us far behind in our work. We fixed problems that appeared to be purely in the application efficiency, by lowering the crash frequency. The same thing can be said for chronic bad-input problems.

A. Keep your hardware in good shape. Go to the "Mr. Goodbyte" session they are having here, and do what they say. If your crashes are still happening weekly or more often, get angry and write a letter setting forth your views on the matter. Maybe HP has been "rabbiting" you, or the people assigned to your account are not going up the chain of command for the resources they need. Find out who the super-guru in your HP office is, and don't bug him if somebody else can answer your question.

B. Leave MPE the resources it needs to do its job, primarily sufficient disc space and table sizes. If you have chronic problems with the spooler shutting down, get mean and tell your boss either less in the way of applications or more disc space or I quit. There is simply no substitute for sufficient disc space, so don't try to fool yourself into keeping things online that prevent you from keeping at least 20,000 sectors free at ALL times.

C. Keep the power clean. If HP says you need better isolation, get it. Don't pinch pennies on the power you feed your computer. Many problems can be traced to this area that didn't look like that at all. (P. S. Deltec wired our isolation unit wrong. If yours isn't putting out the voltages HP says it should, get in touch with Deltec and straighten the matter out. If you don't have a technician who can help you, get a good broadcast radio engineer in for some consulting. This is right up his alley.)

D. Make sure your jobs are ALL restartable from a known point! NOTHING is updated to our data base without preliminary diagnostic reporting to the client. Of course, no program may destroy data it needs to run properly. It should be able to rerun with no problems if aborted at any point! This is hard to sell some programmers, but well worth getting very mean about.

5. Distribute processing.

This term is much mentioned in the industry. I'm not talking about getting a Series 30 for each of your clients and networking. You may already have the hardware you need to take much of the load off your HP 3000. I am talking here about the 2645. This is actually a computer that can run programs loaded via the RS232 connection from HP 3000 disc files. Your 2645 can do input checking and buffering without impacting the HP 3000. You just got yourself a bunch of new computers free!

In far-out cases, consider another CPU. Specs for a POS system we put in called for response time of zero-point-1 (0.1) seconds. Checkers for our cafeteria need to be able to get out a transaction (ie, meal) every 20 seconds on each cash register terminal. No way could the HP 3000 handle that. We loaded up a Cromemco Z-2 home computer with all the memory it could handle, and put in it an extract of our database for validation-lookup purposes. It works fine. In fact, the HP 3000 walks the Z-2 through recovery automatically should the Z-2 crash. The Z-2's program is in EPROM, so it can't be hurt by a malfunction. It can handle a crash on the HP 3000 without data loss, automatically.

If you want to have better maintenence than Cromemco can (by mail), consider an HP 2649. This is the really unbundled version of the 2645, in which you can put together exactly what you want. It could easily be made into a small computer with integral control console. I understand it can even handle a 7920 disc drive if you want that. All this has got to be cheaper than an HP 3000. We would probably have done it this way if we had known what we were in for. If you get replacements by mail, you must keep 100% spares stock plus have your own technicians. That is expensive.

The two factors to look for are:

A. Input editing before submission to the HP 3000.
B. Buffering, to reduce priority demands on the HP 3000.

6. That Bulldog Grip - CONTROL the use of your system.

Always service your existing users well. Don't let their service be degraded by new things. Don't be afraid to tell people "The computer can't handle it." It is very easy to cheat on the numbers (ie, free disc space and system loading) you have learned work well for your system. Don't do it. Spend the time looking for unused capability, and improving the way your existing work impacts the machine, instead.

SLS - A tool to help you do it.

One of the reasons we can do what we do I have saved for a separate section of my presentation. SLS is an acronym for STREAM.LIB.SYS, which is where we keep the program. It is on the swap tape and will we hope make it from there into the contributed library. If you want a copy of it and can't find it, send me a tape and I'll put it on for you.

SLS gives you:

* Documentation of stream files without password problems.

* Documentation in stream files of where variable parameters come from.

* Automatic query of user for variable values.

* Editing and range checking of variable values.

* Automatic generation of items such as current date.

* Stamping of job with information from WHO intrinsic.

* Logic: IF and GOTO constructs, flags.

* Options using UDC's:
  * Initiation of one or more jobs via single UDC command.
  * Save submitted stream for later analysis if necessary.

* (Optional - uses Priveleged mode) Access to session ID, so job can "phone back" to user when it reaches milestones.

* Zoned-decimal editing of parameter data to work with COBOL.

  ** SLS does for job streams what UDC's did for sessions **

How does it work? SLS is a program that reads a file containing statements in a language (see example below) that looks very similar to the MPE Command Interpreter's language. But you can talk to your programs with it, get information from various places and put it on that file, and stream conditionally. The file can be set up with a :FILE equation and a run parameter, so that a single-word UDC can bring up a job by itself.

SLS is versatile enough to be used for other things like data entry in certain very specialized cirumstances. It is powerful enough to reduce drastically the work involved in client-proofing your programs. For more information on SLS, please consult the tape.

Sample job stream initiator file for SLS:

```
DISPLAY SYSDUMP INITIATOR FOR SMC HP 3000-III/768  REV 3.01 1/6/80

<< NOTES:                                                        >>
<<   VARIABLES ARE CREATED BY COMMANDS LIKE FILE, INPUT, SET    >>
<<   AND WHEN REFERRED TO THEREAFTER ARE CONTAINED IN <>.       >>
<<   DISPLAY IS MESSAGE FOR $STDLIST.                           >>
<<   PRINT IS LINE TO GO ON JOB STREAM.                         >>
<<   THIS EXAMPLE DOES NOT USE ANYWHERE NEAR THE FLEXIBILITY    >>
<<   OF THE SYSTEM.                                             >>

<< THIS INITIATOR FILE IS USED BY THE SLS PROGRAM TO PREPARE A >>
<< SYSTEM BACKUP JOB SET.  IT OBTAINS PASSWORDS FROM A FILE    >>
<< CALLED 'IPASS', AND A DUMP DATE FROM THE OPERATOR.  IT      >>
<< STREAMS A JOB WHICH IN TURN SUBMITS OTHER JOBS IF THE FIRST >>
<< STEP (IE, SYSDUMP) GOES OK.  NOTE ALSO THE 'FAKE' OPTION,   >>
<< IN WHICH THE HOUSEKEEPING FUNCTIONS ARE DONE WITHOUT DOING  >>
<< AN ACTUAL SYSDUMP TAPE.                                     >>

<< SET COMMAND IS A STRING VARIABLE ASSIGNMENT.              >>
SET CHANGES NO
SET FILESET @.@.@
SET LISTFILES NO
```

```
<< TODAY SETS A VARIABLE TO DATE, USING SPECIFIED EDIT MASK. >>
TODAY THISDAY MMM DD, YYYY
DISPLAY THE FOLLOWING PROMPT IS TO BE USED IN CASES WHERE WE NEED
DISPLAY A RELATIVE SYSDUMP.  SYSTEM BACKUP OPERATOR MUST REPLY
DISPLAY WITH ZERO UNLESS GIVEN PERMISSION BY THE MANAGER.
<< INPUT GETS A STRING VALUE FROM THE OPERATOR.  WE COULD DO >>
<< EDIT CHECKING ON THIS VALUE WITH SLS, BUT DON'T IN THIS CASE. >>
INPUT DUMPDATE DATE?
MATCH 7 FAKE
<< FILE COMMAND GETS STRING DATA OF A FILE INTO VARIABLES.     >>
<< IN THIS CASE WE ARE OBTAINING PASSWORDS (CAN YOU SAFELY     >>
<< GIVE ME A COPY OF YOUR SYSDUMP JOB STREAM LIKE THIS?)       >>
<< DO YOU HAVE A HARDCOPY OF YOUR SYSDUMP JOB STREAM ON FILE? >>
FILE IPASS.MGR UPS 1 1 8 MPS 2 1 8 APS 3 1 8 GPS 4 1 8 LPS 5 1 8


<< SET UP THE MAIN JOB >>
<< PRINT COMMAND PUTS DATA ON THE JOB STREAM FILE >>
PRINT !JOB SYSDUMP,SYSDUMP/<UPS>.SYS;HIPRI;PRI=CS;OUTCLASS=LP,2
<< LIMIT1C SETS LIMITS TO 1,0 >>
NO 7 PRINT !RUN LIMIT1C.MGR
NO 7 PRINT !RUN ONEJOBC.MGR
NO 7 PRINT !FILE SYSDUMP;DEV=TAPE
NO 7 PRINT !FILE DUMPTAPE=*SYSDUMP
NO 7 PRINT !SWITCHLOG
<< WE PUT DATE STAMP ON LOGFILE AFTER SWITCHLOG >>
NO 7 PRINT !TELLOP ***** <THISDAY> *****
NO 7 PRINT !CONTINUE
NO 7 PRINT !RUN SYSDUMP.PUB
NO 7 PRINT <CHANGES>
NO 7 PRINT <DUMPDATE>
NO 7 PRINT <FILESET>
NO 7 PRINT <LISTFILES>
<< IN CONDITIONAL IF CLAUSES, ALWAYS DO A SHOWTIME OR SOMETHING >>
<< ELSE THAT WILL FLAG ON $STDLIST WHICH PATH WAS TAKEN.        >>
NO 7 PRINT !IF JCW = FATAL0 THEN
NO 7 PRINT !   SHOWTIME
NO 7 PRINT !   TELLOP SYSDUMP HAS ABORTED.  PLACE THE TAPE(S)
NO 7 PRINT !   TELLOP IN FRONT OF MANAGER'S DOOR, AND RUN A
NO 7 PRINT !   TELLOP RELATIVE SYSDUMP AS OF LAST GOOD SYSDUMP
NO 7 PRINT !   TELLOP
NO 7 PRINT !   TELLOP  ** REPLY WITH DATE OF LAST GOOD SYSDUMP    **
NO 7 PRINT !   TELLOP  ** TO DATE PROMPT IN BACKUP.SYS  MM/DD/YY  **
NO 7 PRINT !   TELLOP
NO 7 PRINT !   EOJ
NO 7 PRINT !ELSE
NO 7 PRINT !   SHOWTIME
NO 7 PRINT !   TELLOP ********************************************
NO 7 PRINT !   TELLOP REMOVE WRITE RINGS FROM SYSDUMP TAPES AND
NO 7 PRINT !   TELLOP PLACE TAPES IN VAULT.
NO 7 PRINT !   TELLOP ********************************************
NO 7 PRINT !ENDIF
PRINT !SHOWJOB
```

```
<< FOLLOWING SETS JOBFENCE TO ZERO. SEE SLEEPER IN CONTR. LIBRARY. >>
PRINT !RUN JFENCE0C.MGR

<< FOLLOWING ARE EMBEDDED JOBS >>
PRINT !STREAM,#
<< THIS JOB STREAM DOES A FASTLOAD OF THE NAME-LOOKUP ROSTER FILE >>
<< WHICH ALLOWS PARTIAL-KEY LOOKUPS ON OUR IMAGE DATA BASE. >>
NO 7 PRINT #JOB ROSBUILD,MANAGER/<MPS>.ADMIN/<APS>,DATABASE/<GPS>&
NO 7 PRINT #;INPRI=12;OUTCLASS=LP,2
NO 7 PRINT #PURGE ROSTER
NO 7 PRINT #BUILD ROSTER;REC=-40,19,F,ASCII;DISC=10000;CODE=1
NO 7 PRINT #RUN ROSBLDC;NOCB
NO 7 PRINT #RUN LIMIT5C/<LPS>.LIB.SYS
NO 7 PRINT #EOJ

<< FIND031 DOES DAILY HOUSEKEEPING FOR THE AID AWARDING SYSTEM. >>
PRINT #JOB FIND031J,MANAGER/<MPS>.ADMIN/<APS>;INPRI=9;OUTCLASS=,2
PRINT #FILE NARROW1;DEV=6;CCTL;FORMS=PLEASE MOUNT NARROW1.
PRINT #RUN FIND031C.FINANCE
PRINT #EOJ

<< JANITOR DOES THE DAILY HOUSEKEEPING IN THE SYS ACCOUNT. >>
PRINT #JOB JANITOR,SYSDUMP/<UPS>.SYS,JANITOR;INPRI=6;OUTCLASS=LP,2
<< LISTLOGC PRINTS OUT THE CONSOLE LOG, WATCHES FOR SECURITY >>
<< VIOLATIONS, AND PUTS A SUMMARY OF CPU SECONDS VERSUS INPRI >>
<< VERSUS EXECUTION PRIORITY, ON A FILE. >>
PRINT #FILE CPULOG=LOGCPU.MGR;ACC=APPEND
PRINT #FILE NARROW1;DEV=LP,9;CCTL;FORMS=LOAD 8-1/2 X 11 GREENBAR.
PRINT #FILE OUTPUT=*NARROW1
PRINT #RUN LISTLOGC.MGR
<< WE GET A DISC COPY OF THE REPORT DAILY, IN CASE WE WANT TO   >>
<< TRACK USEAGE CURVES LATER.                                   >>
TODAY MONTH MMM
TODAY DAY DD
ADD DAY <DAY>
PRINT #PURGE RP<MONTH><DAY>.LIB
PRINT #BUILD RP<MONTH><DAY>.LIB;REC=-80,16,F,ASCII;DISC=700,32,1
PRINT #FILE X=RP<MONTH><DAY>.LIB,OLD
PRINT #REPORT @.@,*X
PRINT #RESET X
<< PRCPUC ADDS UP THE ACCUMULATED CPU SECONDS LOGGED ON LOGCPU. >>
PRINT #FILE CPULOG=LOGCPU.MGR
PRINT #RUN PRCPUC.MGR
<< REPFMTS GENERATES A FORMATTED SYSTEM REPORT, WHICH WE DO >>
<< FOR OUR DAILY OPERATIONS LOG FILES. >>
<< THE VERSION OF BASIC (1831) WE USE AT THE PRESENT TIME DOES >>
<< NOT RUN THIS PROGRAM PROPERLY IN COMPILED FORM. >>
PRINT #FILE GREENBAR;DEV=LP;CCTL;FORMS=LOAD 14-7/8 X 11 GREENBAR.
PRINT #BASIC
PRINT RUN REPFMTS.LIB
PRINT
PRINT GREENBAR
```

```
PRINT 2
PRINT >EOD
PRINT EXIT
<< MSGUPDC ADJUSTS THE USER DIRECTORY IN THE MESSAGE SYSTEM >>
<< SO IT AGREES WITH THE SYSTEM DIRECTORY. >>
PRINT #RUN MSGUPDC.MGR
PRINT #RUN MEMLOGAN.PUB
PRINT #EOJ
PRINT !EOD

<< THE FOLLOWING ARE TERMINAL-SERVICING JOBS FOR POS SYSTEMS. >>
PRINT !STREAM CZSYSJ.CAFSYS.ADMIN
PRINT !STREAM CZCARDJ.CAFSYS.ADMIN
PRINT !STREAM TERM49J.CSHOP.ADMIN
PRINT !STREAM TERM51J.CSHOP.ADMIN
PRINT !EOJ
```

This file is an actual copy of the initiator we use for our daily SYSDUMP operation. Note that no passwords are contained in it, and that you can tell exactly where everything necessary to put together a fairly complicated set of jobs came from. This job by no means exploits the full capabilities of SLS.

SLS is written in SPL, and is structured so well by its author (one of our teachers) that several changes have been done to it with no sacrifice in its integrity.

SLS Command Summary

ADD	Adds numbers (literals and/or variables) together
ALPHA	Sets flag true if last input is alphabetic
ANSWER	Sets flag true if answer was positive (ie, Y or YES)
DATECHK	Checks date format, displays message and/or sets flag if error
DISPLAY	Displays a message
EDIT	Edits input data for fixed width, optionally with zoned-decimal for COBOL negative numbers.
FILE	Obtains information from a file into variable(s)
FIND	Search file for string, fetching record number if found
FINFO	Gets information about file into variable(s)
FLIST	Gets sum of EOF's of files named in a file, into variable
GOTO	Transfers control
IFERR	Sets up trap for file errors (see ONEND in BASIC)
INPUT	Inputs a variable, prompted
JSNUM	Puts job/session ID of submitter into variable
LOAD	Loads a variable into the input buffer
NO	(May precede any other statement) Do if flag negative
NULL	Sets flag if last input was null (CR only)
NUMERIC	Sets flag true if columns indicated are numeric
PRINT	Prints information on file to be streamed
RANGE	Sets flag true if last input within specified range
REPEAT	Go back to last INPUT or ANSWER
SET	Assign value to variable
TODAY	Get date information into variable
WHO	Get information from WHO intrinsic into variables
YES	(May precede any other statement) Do if flag is true

More complete documentation is contained in the file SLS.DOC.SANJOSE.