IMAGE/3000 - The Cost of Single Threading in a Large Data Base.

William M. Lyons
Senior Technical Analyst
GTE Data Services, Inc.

HP - Higher Performance? Yes. HP continues to produce larger and faster computers. However, as with anything man-made, weaknesses can and do exist. When developing or even maintaining applications on the HP 3000, it is vital to keep these weaknesses in mind or the applications become vulnerable through these points. For the HP 3000, one of its strongest points even has a weak point. IMAGE/3000 was developed in a time of slower CPUs and slower disc drives with only one controller. Even the maximum number of terminals that could be connected to the system was much lower and the baud rate for terminals was 2400 not 9600 or 19200, now possible with the HP150 and ATPs on a Series 64. This paper is not intended to fault IMAGE/3000, it is just to make all users aware that there are problems that can be overcome in our applications, since HP has stated, previously, it would require the rewriting of IMAGE to correct them.

Today, I will discuss our reasons for being concerned with performance, our findings through our investigation, and the solutions we have implemented or considering to implement. Those involved in the testing included Dorann Barenbrugge of GTE, Hugh McKee and Mark Conroy both of the HP Tampa office.

I work in the Information Systems Group where we develop application systems to meet the needs of the telephone operations area of GTE. Our applications are developed to serve the common needs of all the various telephone companies in GTE. Some companies are very large while others are small. They differ from company to company on needs and we have to develop a core system with flexiblity. The specific project I work on is called CNAS, Circuit Network Administration System. About ten years ago, this area was mechanized using an IBM mainframe with sequential master files that were updated once a week. This system recorded the makeup of circuits which connect the various central offices of the phone company. It also recorded the makeup of Special Service Circuits (leased lines, conditioned lines, extended local area service, etc.). Some of the transactions for this system were quite large due to the amount of data required to describe a circuit or its facilities. The result was a lot of grief when a transaction of 24 cards long had 1 character punched wrong. To eliminate many of the simple errors, a front end data collection system was developed on the HP 2000. Weekly, the transactions were extracted to tape and processed through the IBM batch system.

The overall combination of the IBM and HP 2000 made the data smoother to enter because the HP 2000 pointed out simple edit errors on input. However, it did not make the data accessible on demand. Therefore, we were authorized to develop an online system to replace the original IBM/HP 2000 system. After an investigation into several CPUs, the HP 3000 was chosen as the host system. The development took about two years and was installed in the first telephone company on a Series III with seven 7925 disc drives, one of which was a serial disc drive for backup purposes. More terminals were connected than originally estimated. As other departments found out that they could retrieve data useful to them, they requested access and a terminal. Also, the users found that data not originally anticipated could be stored in the data base. This added up to more than the system could handle. More disc space was added and more memory, too.

The Series III was monitored by our Hardware/Software Evaluations group and found to have a bottleneck on CPU power. At the same time HP was introducing the Series 44. One was ordered to replace the Series III and arrived about one year after the application system went live. In the same time frame, MPE III was being replaced by MPE IV. The combination provided the extra power needed. With the Series III, 21 - 22 concurrent sessions was the maximum before response times would deteriorate. With the new Series 44, no problems were noted until more terminals were added several months later. The limit now seemed to be about 35-37 sessions.

About this time, we became more involved with the performance investigation. The first thing we did was to monitor the production system with OPT/3000. We did this using both batch and online observations. The batch mode was set up to report every hour and print once a day. We found the system to be disc I/O bound. The CPU was busy 52% of the time while it waited on disc I/O 27% of the time. The disc I/O rate was only 34.6 I/O's per second. With two controllers, this seemed low. This prompted us toward looking at why and eventually led us to the Global Data Base Control Block.

Each user has a local control block for the data base for each DBOPEN but only one global control block exists for a data base and only one user can perform an intrinsic call against the data base at a time. This concept is called Single Threading. For example suppose three users are accessing the same data base. User A wants to do a DBPUT, User B wants to do a DBUPDATE while User C wants to do a DBPUT also. They would queue up as follows:

1. A would take ownership of the global data base control block.
2. All the disc I/O and other work required to place the record would be executed prior to releasing the global control block.
3. B would be able to take ownership and execute its DBUPDATE releasing the global control block after it is finished.
4. Finally C would get the global control block and execute its DBPUT.

Every user must funnel through this control block one at a time to execute an IMAGE intrinsic call against the same data base.

Among other observations, we saw that most of the users were using the data base for online inquiry or producing printed reports in a Query job. This proved to provide a solution later. However, the new Series 64 was being introduced. We did some testing in the Atlanta HP Sales Office to first insure compatiblity with the application programs. (We were in development before VIEW came out and developed our own terminal I/O routines.) We also checked the performance of the Series 64 and to obtain a comparison, tests were repeated on the Series 44 and Series III systems in Tampa. All three systems were similar. The Series 64 had two 7933 disc drives. The Series 44 had ten 7925 disc drives. Both of these systems had two masters while the 44 had additional slave drives. The Series III had one master 7925 disc drive and three 7925 slave disc drives. The Series III had look ahead seeks enabled. The test showed that there is an increase in capacity, until IMAGE, with one data base, was introduced.

The following measurements support these findings. The measurements are split into two parts. Part I deals without IMAGE. Part II deals with IMAGE.

I. Performance using an evaluation program.

The program was written by the Hardware/Software Evaluation group of GTE Service Corp. It allows the user to vary the load forced upon the system by writing to disc, tape, printer and punch card devices or CPU usage. For the purposes of our evaluation, five measurements were taken. Disc I/O rates were those observed using OPT/3000.

A. CPU only -

Job with 450,000 loops through a series of arithmetic equations.

	III	44	64
Elapsed time (sec.)	1087	687	233
CPU time (sec.)	1038	675	231

This is the strong point of the Series 64, being about three times faster than the Series 44 and about five times faster than the Series III.

B. Disc I/O - One drive only

Job with 10,000 physical writes to the system disc.

	III	44	64
Elapsed time (sec.)	246	245	239
CPU time (sec.)	106	72	40
Disc I/O rate per second	40.5	41.4	41.3

Again, the CPU is faster, but, the elapsed time barely dropped from the Series III and Series 44. However, this was using only one disc drive and one controller.

C. CPU and Disc I/O together -

Job with 10,000 disc writes using a single disc drive and 100,000 loops through the same arithmetic equations.

	III	44	64
Elapsed time (sec.)	594	400	236
CPU time (sec.)	543	361	142
Disc I/O rate per second	17.1	25.2	42.9

This points out that because of the faster CPU, a mixture with CPU and disc will spend less time on the CPU dependent activities and be allowed to process more disc I/O.

D. Disc I/O and CPU-Heavy

Two jobs running concurently that were used for parts A and B.

	III		44		64	
	A	B	A	B	A	B
Elapsed time (sec.)	1152	1259	713	850	239	432
CPU time (sec.)	1037	105	676	71	232	40
Disc I/O rate per second	4.0/41.0		3.7/41.4		3.8/42.1	

The Disc I/O rate shows as the lower figure when both jobs were competing on the system. The higher figure shows the rate that the disc I/O job achieved after the CPU bound job completed. Again, this points out a very fast CPU but disc I/O to a single disc drive about the same for the Series 64.

E. Disc I/O - 2 drives

Two jobs running concurrently that were the same as in part B but going to two different drives. On the Series 44 and Series 64, the drives chosen were on different controllers. Job A below is for writing to the system disc; Job B is for writing to another drive.

	III		44		64	
	A	B	A	B	A	B
Elapsed time (sec.)	402	394	252	473	266	239
CPU time (sec.)	99	99	73	70	40	40
Disc I/O rate per second	49.0		62.5		84.7	

This shows the improvement in disc I/O rates that could be achieved in the upgrading of the hardware. In fact, our further testing on another series 64 with four 7933 disc drives and four jobs writing concurent to separate disc drives pushed the disc I/O rate to 116.6 per second.

II. Performance through the application programs using IMAGE/3000

Another phase of testing on the HP 3000 Series 64 and Series 44 depict the way the application will work. This was accomplished by eliminating operator think time through setting up the online programs to run in batch mode. To do this, the terminal I/O routines were substituted with routines to read input from files on disc. The modification was only a small amount of code compared to the total program in each case. Within the read routine, a time stamp was established to give us response times for the various transactions. The system was again observed, during these runs, using OPT/3000.

Several measurements were taken.

A. Each online mainline was run, one job per mainline, for a total of eight jobs. Activity was against the system disc only. Listed below are OPT/3000 observations:

	44	64
CPU Busy	24%	15%
CPU Paused for Disc	72%	82%
Disc I/O Rate Per Second	29.1	30.7

Transaction	Response time/CPU time in seconds	
Simple type 1	18.1/00.279	28.3/00.122
Simple type 2	6.6/00.114	2.9/00.060
Heavy I/O type 1	433.2/22.455	479.1/12.083
Heavy I/O type 2	603.7/81.308	669.6/47.571
Medium type 1	24.5/00.761	39.6/00.369

Total by job	Elapsed/CPU time in seconds	
1.	153.1/003.821	203.3/01.818
2.	1585.9/182.214	1686.7/98.060
3.	1449.9/178.620	1466.9/98.616
4.	319.8/012.663	352.9/06.471
5.	142.8/004.620	187.2/02.069
6.	31.4/001.405	63.6/00.636
7.	134.4/018.261	332.5/09.270
8.	161.7/005.410	141.0/01.926

The above results show again the much faster CPU of the Series 64, but elapsed times not any better than the Series 44. The figures show worse times for the Series 64 in some instances. There may have been a bias to the test as job 6 dropped out prematurely for an unknown reason. The overall effect shows that the Series 64 with IMAGE could get about 30.7 disc I/O's per second as opposed to the Series 44's 29.1 I/O's per second.

B. The same test was done again but now the Master and Detail data sets were split across disc drives and controllers.

	44	64
CPU Busy	24%	15%
CPU Paused for Disc	67%	78%
Disc I/O Rate Per Second	29.2	35.9

Transaction	Response time/CPU time in seconds	
Simple type 1	25.8/00.270	17.4/00.118
Simple type 2	5.2/00.120	5.9/00.052
Heavy I/O type 1	495.7/23.048	396.7/12.526
Heavy I/O type 2	761.7/81.073	630.2/47.463
Medium type 1	33.3/00.732	28.5/00.397

Total by job	Elapsed/CPU time in seconds	
1.	181.7/003.800	149.6/01.794
2.	1928.9/182.586	1528.8/98.210
3.	1756.1/178.674	1400.8/98.475
4.	347.7/012.870	282.3/06.637
5.	166.4/004.640	140.9/02.247
6.	107.1/005.370	102.3/02.574
7.	152.5/018.307	121.9/09.371
8.	190.8/005.535	161.6/02.784

This configuration, with two disc drives, is closer to the model of a typical site for our application. The results show that with a Series 64, improvement can be expected. However, this improvement is small compared to what had been hoped with upgrading to a Series 64.

C. This test consisted of two parts. The first had eight jobs adding and deleting records through the programs in such a way as to cause heavy IMAGE activity. All activity was against a single data base. The observations were as follows:

	44	64
CPU Busy	25%	15%
CPU Paused for Disc	70%	83%
Disc I/O Rate Per Second	29.9	30.9

The second part was the same as the first part except four data bases were used instead of one. There were two jobs per data base. The observations were as follows:

	44	64
CPU Busy	34%	20%
CPU Paused for Disc	55%	73%
Disc I/O Rate Per Second	44.6	58.3

This shows that the systems are capable of doing more disc I/O. Each data base has a global control block which allows only one IMAGE Intrinsic to be satisfied at a time on a system level. With four data bases, there were four global control blocks, and four intrinsics calls were being satisfied at the same time. For the Series 64, this meant almost a 100% increase in the disc I/O rate.

D. To further confirm the bottleneck with the Global Data Base Control Block, this test was conducted with two jobs in part A. Both were heavy on the IMAGE calls for disc I/O. They were accessing totally independent data sets in the data base. The first part again used only one data base.

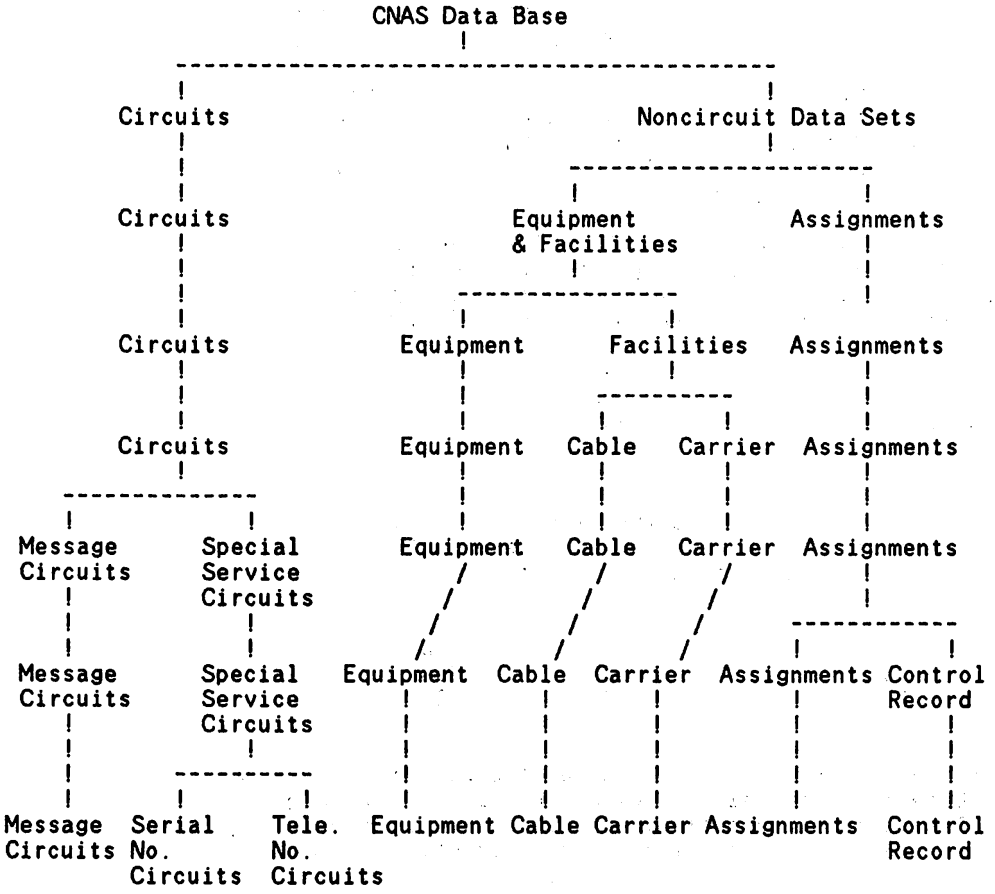
	44	64
CPU Busy	36%	13%
CPU Paused for Disc	58%	85%
Disc I/O Rate Per Second	26.3	30.0

Submitting the same jobs against separate data bases gave the following results:

	44	64
CPU Busy	35%	18%
CPU Paused for Disc	58%	78%
Disc I/O Rate Per Second	38.6	44.0

Again, this shows an increase in the disc I/O rate.

With this information, it became clear that more than an upgrade to the CPU would be required. We developed programs to simulate the transactions of the application, which would also allow the data sets to be moved from data base to data base as needed for the splitting. We also took our original data base schema and developed a logical breakdown of the data sets first into two data bases, then three, four, five, six, seven, and eight data bases. In some cases, an automatic master was duplicated in two separate data bases in order to place two detail data sets pointed to by the automatic master in separate data bases. Each level below is a depiction of the breakdown.



The following is a synopsis of the various schemas and the change in space requirements.

Number of Data Bases	Number of Data Sets (Root Files/ other)	Number of Paths	Total Disc Sectors	% Increase from One Data Base
1	1 / 78	138	300,587	-----
2	2 / 78	138	300,595	.0266
3	3 / 79	138	297,546	-1.0116
4	4 / 80	138	302,441	.6167
5	5 / 81	138	306,830	2.0769
6	6 / 90	146	305,784	1.7289

7	7 / 90	146	305,790	1.7309
8	8 / 104	164	304,659	1.3546

In a few cases we also split detail data sets that were shared between categories. The capacities were adjusted to handle the same amount of data and as the one column indicates the space required stayed about the same.

Our testing was done on a Series 64 with four 7933 disc drives. The test was conducted but the results were somewhat unclear. Therefore the tests were repeated with more tools on two separate occasions. Our local SE's were involved in these tests and the two repeat sessions.

The first phase of testing was observed using OPT/3000 with 16 jobs running in the background. It was run and set to produce summary reports for each minute of run time. In all cases, the reports showed an initial load time that involved loading of the programs, opening the data bases, and other activity. Eventually, the figures became fairly stable. The following table represents an average of the stable figures:

Number of Data Bases	CPU %Busy	CPU %Paused Disc	Disc I/O Rate (per Second)
1	19.4	77.5	24.8
2	30.3	61.8	44.0
3	31.0	60.7	44.0
4	47.0	39.5	56.0
5	72.0	11.8	45.6
6	60.5	22.8	52.3
7	62.2	22.2	49.6
8	64.3	19.3	54.7
*0	63.3	11.3	116.6

* This measurement was taken through a program not using IMAGE to establish the maximum capacity of the system involved. Four jobs were run concurrently with each writing to a different disc drive.

It was anticipated that, as the system disc I/O rate increased, the CPU would become busier and that there would be a smooth curve produced by the figures. However, there was an unexpected sharp jump in CPU busy and drop in the disc I/O rate for five data bases as opposed to four data bases. The test was repeated that same day to confirm that no unknown influence had biased the test. The same results were displayed in the reports from OPT/3000. Our local SE's, who assisted in much of this endeavor, met with us to review the results but were not able to determine any reason for the drop. We again repeated the test, so they could monitor it with APS/3000 (Sampler). It was hoped that we could pinpoint in what code the jobs were spending more time with five data bases. The test was conducted with the same program that was used before. This time, to give a contrast, the test was conducted with four and five data bases with twenty and eight jobs running. The following are the results of that test:

Number of Data Bases	Number of Jobs	CPU %Busy	CPU %Paused	Disc I/O Rate (per second)
4	8	68.0	15.0	53.4
4	20	56.0	27.2	56.8

5	8	76.0	10.0	39.8
5	20	74.4	6.8	51.4

The monitoring with Sampler provided very little insight. It showed no place in the code that stood out as taking most of the CPU time. The extra CPU time that was required by five data bases was evenly spread between all code segments. There was a slight increase in one of about 4%. All of the other segments were less.

After further consulting among the HP software personnel, locally and elsewhere, the test was repeated again. This time, the system was monitored by OPT/3000, APS/3000, and MPE Data Collection. The time of observation was extended from ten minutes to thirty minutes. Also, thirty-two jobs were used with one, four, five, and eight data bases to draw a comparison. The results were as follows from OPT/3000.

Number of Data Bases	CPU %Busy	CPU %Paused Disc	Disc I/O Rate (per Second)
1	24.0	71.0	26.8
4	58.3	20.8	56.1
5	73.8	4.0	52.8
8	66.0	8.0	64.0

The results were again inconclusive as to why there is a change between four and five data bases. Sampler and MPE Data Collection did not show anything. The files of data collected were passed onto senior technical people in HP, but, nothing has been resolved, to date. Therefore, it was concluded that the change we would make had to be flexible.

We learned how to manipulate the data base into multiple data bases.

1. Using Query, produce a "Form Sets" listing. Number the data sets in sequence. The root file name plus this number gives the MPE file name for that data set.
2. Draw up new schemas using the capacities shown in the "Form Sets."
3. Store the data base to tape and then purge the data base using DBUTIL.
4. Create the root files by using DBSCHEMA and the new schemas.
5. Create the data bases using DBUTIL and the root files. This is required because the root file has an indicator that shows whether a data base has been created or not.
6. Using Query, produce a "Form Sets" listing for each data base.
7. Using FLUTIL3, change the root files from PRIV to NONPRIV files. (File code for the root files are -400 and other data base files are -401. The file code can be changed to 0.)
8. Using DBUTIL, purge the new data bases. The root files not marked PRIV do not get purged.
9. Using FLUTIL3, return the root files to PRIV.
10. Restore from tape the data base files, except for the root file.
11. Using FLUTIL3, change these files from PRIV to NONPRIV.
12. Using the new data bases' "Form Sets" produced in step 6, draw up the MPE file name for the data sets, using the root file name and sequence of the data sets.
13. Rename the data sets as required to move the data sets between the old data base and the new data bases.
14. Change all the data sets back to PRIV files, using FLUTIL3.
15. Recheck your steps. If possible do this on a printing terminal or one with an integral printer and log everything. Also do it with a few people double checking your work by watching you

over the shoulder. It is very open to problems and should be done with the utmost of care.

We concluded from our work, that it is better not to split paths and create duplicates on any data sets including automatic masters. However it can be done. Before storing the data base to tape, remove paths that will be split. Create the schemas without these paths. Then, readd the paths, after step 14. We found that using Adager to delete the path and readd it was fine for our test data bases. However, some of the production data bases which have about 3,000,000 sectors involved make it difficult. Also, to make it flexible to meet the needs of the various companies we couldn't duplicate the name of of a data set.

The solutions are as follows:

1. The Series 44 was kept as a backup system for the application and noncritical additional applications. The data base is stored to tape every night from the main system, a Series 64 and restored to the Series 44. The Series 44 is used for listings only, by those who can use the data from the previous day rather than the current data base. It also is used for query since the data is never more than 24 hours old and most of the inquiries did not need any more current data. If the up-to-date data is required they can shift to the Series 64 and repeat the inquiry.

   This same procedure could be done on the same system using GETFILE2 to restore the data base into a separate group or account if needed. The result is two global control blocks for the two data bases which will improve the retrieval time. However, twice the disc space is required.

2. Split the data sets between data bases and thus allow more con- current activity. If you are just designing the application system, this will be fairly easy. Keep it flexible because activity will probably vary from what is anticipated. If you have a system already developed you may need to alter your programs. Our plan is as follows:

   A. Divide the data base as above without splitting paths. In other words any data sets that are linked by a path or series of paths will not be split into different data bases. B. Alter all the programs to call an indexing subroutine. This routine would be passed the data set name and would pass back the data base name to be used in the subsequent IMAGE intrinsic call. This routine would also issue a DBBEGIN when the intrinsic call is the first for a given data base and the call is for DBPUT, DBDELETE, or DBUPDATE. This routine should be very efficient in looking up the data base name. In our case each data set has a numeric name and thus could serve as an index into a table. C. The calls to DBBEGIN are removed from the programs since the indexing subroutine will take care of this. Where DBEND is called now, a subroutine would be called to check a table of the data bases and issue DBENDs for all the data bases that a DBBEGIN was issued against. This is called stacked DBENDS. This limits the window that a transaction could be marked complete for one data base and not for the other data bases involved. Unfortunately IMAGE does not allow transactions to be marked as covering multiple data bases. A program to analyze the log file could be written to change DBENDs to abnormal DBENDs. This may not be needed because the window for problems is so very small. Only a system failure in the middle of a stack of DBENDS would cause a problem. D. We planned to monitor the effect of splitting 4 and 8 ways with the production data base at a phone company. The routine in part B would be issued to all phone companies with the option of having 1, 4, or 8 data bases involved. E. There would have to be tighter controls over backups to keep all data bases in sync. Recoveries from log files and restores from backup would also need very tight controls. Perhaps HP can assist us by providing a way to define a transaction accross multiple data bases and have DBSTORE/DBRESTOR handle multiple data bases on the same set of tapes. This would eliminate a great deal of the uncertainty of multiple data bases.

   These are still only plans. Due primarily to the promises of HP in the area of Disc Caching approval to go ahead with this change has not been given. The change for us is quite costly due to the size of the application. The decision is to wait until measurements can be taken from Disc Caching. However, Disc Caching may not provide the performance improvement needed to large data bases and it is anticipated we will eventually be given approval.

We could not find instances of simple data entry in our application. However, it is possible that the use of message files that would be added to online and read by a batch program

could speed up the entry work. The batch program could do the actual DBPUT. We do not have figures from testing this but it has been suggested as a way to improve performance and it seems logical. However, spliting the data base is the most realistic way to improve performance.

Splitting the data base has some other advantages. First, IMAGE has maximums for the number of data sets, items, etc. that a single data base may have. By splitting the data base there is more room for growth before these maximums are reached. Second, as it stands now, the data base is too large to go through a normal DBUNLOAD and DBLOAD to correct broken chains or reorganize the data base. It was done one time and took about a week to complete the whole process. With smaller data bases, individual data bases could be done on weekends. As it stands now, to correct broken chains a person has to do it by hand using DBDRIVER.

To sum this up in a very few words, IMAGE/3000 is better used as a handler of multiple data bases than of one giant data base. Large applications should not depend on IMAGE to keep the data in separate data sets in sync by keeping the data sets in the same data base. When this happens the response time will increase and the disc I/O rate will decrease.

In essence, do not keep all your eggs (data sets) in one basket (data base).

William M. Lyons is a Senior Technical Analyst for GTE Data Services, Inc. in Tampa, Florida. He has worked on applica- tions on the HP 3000 for over six years. A native Floridian he graduated from Florida Atlantic University in Boca Raton with a degree in Mathematics, and a minor in Computer Science. He also holds a Masters degree in Mathematics from the same institution. Currently, Bill is serving as the Vice Chairman of the HP Florida Regional Users Group, FLORUG.
