Paper 120-25

Minimizing Development Risk While Maximizing Warehouse Performance on UNIX® Systems

Jeff LeSueur, BMG Direct, New York, New York

ABSTRACT: Like many companies, BMG Direct is coping with business challenges which put pressure on sales, profitability and overhead. While a data warehouse represented considerable long term potential in developing incremental profit, it also represented a significant and immediate expense and risk. BMG management needed an approach which assured that development would be both successful and within budget.

By adopting a low risk approach with management, which emphasized cost neutrality and a "try before you buy" test effort coordinated with SAS and IBM, initial funding was easily acquired to support a bench mark test of the warehouse hardware and software. Testing would confirm the design goals could be met, and would expose any unexpected risks. Since actual performance was nearly four times better than expectations, the project was approved with a high level of confidence.

This paper focuses primarily on the design and test aspects of warehouse development and documents the activities and key items learned in the course of successfully developing a large (100GB), high performance data warehouse. IT managers and business managers should gain insight into system performance parameters available from UNIX based minicomputers, as well as low risk project management guidelines.

BACKGROUND

For several years all analytical activity at BMG was performed on either a production IBM S390 system, or one of two small (single processor) UNIX systems. Data on the mainframe was stored on 3490 magtape (2GB capacity). The production masterfile had grown to 19 such tapes, DASD storage for this large a file was simply unavailable, even sort space exceeding 1.5GB was normally unavailable. Production masterfile data was also routinely 'purged': older, inactive accounts were moved to another set of tapes, 15 or so comprising purged data from the prior 18 months.

To improve analytical response times, a 5% sample of the masterfile was maintained on 3 (low density) 3490 tapes. While this was generally adequate for test and sample based analysis, it was refreshed weekly (random select) which frustrated analytical continuity, observation volumes less than 1,500 (total) suffered from sample error, and the data was inherently not shareable at one time (tape format) which led to occasional job queuing. In addition, mainframe CPU cycles were becoming increasingly less available: jobs which used to run in about an hour were suddenly taking up to three hours to complete due to "production priorities".

These constraints became a major impediment to expanded analytical activities. In parallel, business goals were being increasingly tied to successful analysis of larger and larger amounts of data. A comprehensive and highly responsive database on disc was clearly necessary.

Production priorities mitigated against expending funds for expanded mainframe resources: we would pay 100% for the expansion and likely get far less than 100% of the performance improvement. The data volumes were well beyond the capacity of most Intel based PC Servers. Previous experience with small UNIX platforms was very good, but the data volumes managed on these systems were relatively small compared to BMG's complete customer masterfile, and internal knowledge of current minicomputer performance standards was exceedingly limited. We simply did not know if we could manage 100GB of data on a minicomputer, or if we could refresh 50GB every weekend, including indexes, reports and backup.

WAREHOUSE PLATFORM DESIGN DEVELOPMENT

To assist in developing potential solutions, our NYC SAS Account Manager, Mark Barry, arranged for preliminary discussions to be held with Margaret Cravar of the SAS Systems Solutions Group, and the IBM technical liaison. The barest of outlines was discussed over the phone, no more than that provided above. They recommended the IBM F50, a midrange (two to four processor) RS6000 based system with capacity for over a terabyte of SSA disc storage.

Sam Gironda of IBM RS6000 Marketing put us in touch with Al Schwarz of the Ergonomic Group. Al generously arranged for several conference calls with IBM and IBM-industry systems engineers to further clarify system performance. It was quickly pointed out that performance on this application would largely be determined by disc I/O performance. SSA disc performance, being multi-ported and asynchronous, offered higher performance than SCSI, which is synchronous (daisy chained). The upper limiting factor on SSA was a combination of disc data rates (approximately 20GB per hour, dependent on disc size) and the SSA controller throughput of 70GB per hour (35GB on each of two channels).

Based on these times, a database development timeline confirmed that the F50 system with several hundred GB of disc space would likely support our requirements. Based on considerations of long term scalability and price/performance, the F50 was designed out in favor of an S7A system, which offered configurations of 4, 8 or 12 processors. This was with the expectation that project scope would expand if the initial database was successful.

It may be helpful to clarify "timeline" referred to above. This is the sequence of major database functions actions required in refreshing the warehouse over a weekend.

Load Raw Data from Tape (EBCDIC)
Create SAS Data Sets
Create Sample Data Sets
Create miscellaneous Data Sets
Index all Data sets
Backup Database
Execute Reports

Each of these actions was estimated and a timeline developed. The goal was a maximum of 32 hours, the initial estimate was 23 hours, providing a reserve of 9 hours. This was not a significant reserve, given the potential variability of the estimate and operational variability. Firmer knowledge of performance , or higher performance than expected would be very beneficial.

DATA MIGRATION FROM PRODUCTION

At the time, and continuing until the benchmark test in Dallas several months later, the unanswered question was how to move the data from the production system to the minicomputer. The source data was stored on 20 3490 tape cartridges, in EBCDIC. IBM Infospeed, a high speed gateway, was a possible solution, but not available to us. Minicomputer tape standards are generally 4MM and 8MM, not 3490 tape cartridges. IBM offers both the 3490E (2GB) and 3590 (20GB) capacity tape drives on the RS6000 series. As noted above, the source data was already on 3490 tapes, written on the MVS system. If the MVS tapes could be read on the UNIX 3490E, then standard SAS EBCDIC informats could be used to convert the mainframe tape based data to SAS datasets, *provided* the tape throughput plus the conversion overhead was within the overall warehouse development timeline.

BENCHMARK TESTING

To confirm that the 3490 tape based source data could be read on a minicomputer, and that the performance requirements for reading the data and converting it to SAS datasets were achievable, benchmark testing was discussed as a possibility. SAS vendor relationship management was critical in arranging this final step in development. IBM arranged to provide a system and support for 5 days, as well as the IBM RS6000 performance monitoring hardware and software. Mark Barry, our NYC SAS account representative, further coordinated the availability of suitable technical support from Leigh Ihnen and John Sims of SAS Systems Development.

BMG management quickly approved the nominal investment in testing. With a hardware and software configuration in hand, and an approved budget adequate to the same, arrangements were made to benchmark the performance at the IBM RS6000 Benchmark Center in Dallas, Texas.

The benchmarking effort described was not undertaken as an evaluation of alternative configuration implementations. It was taken as a given that the highest (practical) performance would be required to refresh the data every weekend, between 12:00 Midnight Saturday and 8:00 AM Monday. Benchmarking was intended to demonstrate that all stages of the data refresh timeline – data transfer from tape, creation of SAS datasets, indexing, backup to tape, batch reports – could be performed within the 32 hour goal. Scalability was expected to provide growth to match end user performance requirements.

Testing would proceed against a test plan prepared by BMG which called for reading and writing increasingly large portions of the Master File, to determine if the performance/volume relationship was linear, and therefore predictable. This would also help identify where and why performance deteriorated, if that were to be the case. The end goal was to ensure that the complete Master File could be read/written as SAS datasets within 10 hours, the allotted time available for creating the warehouse data structures during a weekend.

The master file was 34.5 gigabytes (GB) in size, with approximately 50% of the Master File record volume contained in the first 12 of 19 tapes. The major uncertainty in the test was the ability to read the Master File Tapes. The Master File contains many fields in IBM packed decimal format. This necessitates maintaining the data in EBCDIC when transferring to UNIX (or any other platform) for correct translation of packed decimal on the target machine. However, without a 'handler' to convert EBCDIC to ASCII, the tape content.. was visually unreadable on the UNIX system. Hence the uncertainty if the data transfer from tape was at all successful.

The use of packed decimal fields is a troublesome issue: packed decimal conserves record space but restricts the ability to move the data off the source platform. This is particularly vexing when it is noted that EBCDIC to ASCII conversion is standard on TCP/IP and other communication software. On the other hand, even an electronic transfer of ASCII data would require conversion to SAS dataset form.

To eliminate the possibility that the UNIX 3490 tape driver could at least read a tape from start to finish, a sample tape was forwarded to Don Ferndelli at the benchmark facility. While he reported a successful data transfer, the visual content, being EBCDIC, was indecipherable. This step at least eliminated the possibility of a system level incompatibility.

BENCHMARK TEST PLAN

Largely through the insistence of Al Schwarz, a very detailed test plan was developed. This plan included step by step required inputs, processing, and expected outputs. Thinking ahead to that level of detail was difficult, yet Al Schwarz was merciless in his insistence that, without specific actions reasonably timed to occur with the five available days, we risked not accomplishing the primary performance test goals.

The level of detail, particularly the time scale, proved very useful on several accounts: testing was structured to fill the available time (5 days), and the schedule demonstrated how this was possible. Priorities were arranged to put the most important test goals first in the schedule, and to ensure time to handle problems. Because of the detailed schedule, we were generally confident that all planned activities were achievable throughout the testing. (Of course it helped that throughput rates were 50% faster than expected.) Once on site, the plan provided everyone with full knowledge of “what do we do next”. And as Al pointed out several times, if a problem had developed, we would have been known immediately what following steps would be impacted or skipped entirely, while the problem was fixed.

The following summarizes (somewhat) the test plan detail:

1) **Day 1:** Resolve open issues regarding disc storage (RAID, Mirrored, Stiped.
2) Confirm AIX (operating system) parameters (file system, large file implementation, striped configuration)
3) Install File System, establish agreed upon system parameters
4) Confirm all hardware was addressable and working (disc read/write).
5) Load SAS software, confirm system parameters, defaults, large file implementation, system options
6) [LUNCH]
7) Test execution of base SAS program (simple data step, proc print, log and output window activity.)
8) Load SAS benchmark program (essentially a long input statement) and preliminary test program (short input statement).

9) Test transfer of data from tape to disc, reading several fields with preliminary test program to confirm EBCDIC/ASCII conversion.
10) Test reading sample set of tapes and confirm complete readability through random sample of entire dataset using complete input program (all 260+ fields).
11) Confirm content of resulting dataset (observe record detail, xtab several fields with known distributions. Examine all input field errors.)
12) Evaluate data timing of tape read / write and dataset creation against base timing, confirm test plan timing figures.
13) [Dinner]
14) **Day 2:** Load all mastefile tapes.
15) Test timing of converting 100,000 records, 200,000, 400,000, 600,000, 800,000, 1,000,000 records, confirm linear performance, estimate time to process complete file.
16) Test timing of converting 2,000,000, 4,000,000, 8,000,000, 16,000,000 records (complete file, overnight execution).
17) **Day 3:** Confirm completion of overnight job execution, resulting datasets on 20MM customer records. Analyze dataset sizes, review file system space allocations.
18) Evaluate dataset conversion times for performance fluctuations related to volume of data or speed imbalances between CPU and disc I/O. Adjust buffer space accordingly, AIX read-ahead parameters, other parameters affecting resource balance.
19) Evaluate conversion time against expectations and master timeline.
20) Evaluate dataset space and allocation of file system space. Initial file system was structured based on Work Space, Staging Space (Tape input data), Target Space (Final datasets, including indexes and sample data). Confirm correspondence to plan.
21) **Day 4**: Test Index processing time. Test Indexing performance on various file sizes. Evaluate index timing against planned timeline.
22) (3590 drive was not available for testing backup process).
23) Test/Simulate multi-user query sequence. If time permitted, a simulation of query loading would be performed by testing a simple query, then repeating it for several tables, comparing degradation of performance. (With some irony it was pointed out that each query would have to be against different tables, as simply repeating the same query would result in memory and disc cache hits and a corresponding *improvement* in performance per query.)

DESIGN AND PERFORMANCE RESULTS

DISC STORAGE: 'Straight', Mirrored, RAID, or Striped: As noted above, the disc storage issue was resolved in favor of Striping in 6 * 64K blocks, sometimes known as RAID-0. It was decided that protection provided by RAID 5 (checksum) could almost as easily be provided by a simpler tape backup. Striping would improve system throughput and conserve disc space. Data rates and storage size provided by the 3590 tape device were surprisingly high, providing fairly rapid restore possibilities.

Disc striping entailed storing data across a range of discs, instead of just one. Striping generally improves throughput: data can be read/written from several discs at once, in a suitable fraction of the time it takes to read the same quantity from a single drive.

An option existed to stripe some file systems and mirror others. However, the expected significant increase in throughput would affect all aspects of the warehouse: work space, source data write/read, as well as final SAS dataset creation. Because the throughput increase would be so significant, and the expectations for disc reliability are high, data striping of all data spaces was performed. Striping of the operating system space, located on SCSI disc, was **not** performed. In fact, to improve system reliability and restoration time, the operating system space was **mirrored** to a second SCSI drive installed after the final system was in place.

For real time transaction based systems, striping of data spaces would likely be unacceptable, as loss of any data sector or drive would take out an entire file system, with no recourse but to replace or re-map the drive, and restore the *entire* file system from back up tape. Downtime would be as much as a full day. This was acceptable for us, and our expectations were made accordingly.

It is my understanding that this risk could be mitigated by mirroring the data in conjunction with striping. Mirroring entails writing data to two drives simultaneously. In the event of a failure, the duplicate drive is expected to come into use immediately. However, the cost of duplicate drives for this application could not be justified

WAREHOUSE DATA STRUCTURE (SAS Data sets)
With several years experience responding to analytical issues, the paramount consideration was rapid access to more recent sales transaction history. Marketing wanted to know what happened last week and Finance wanted to compare last week to the same time last year.

Since the primary use of the system would be in providing marketing information and developing behavior models, access was optimized as follows:

CURRENT CUSTOMERS
- Prior 12 Months Sales
- 12-24 Month Sales
- 24-N Month Sales

Non-Current Customers
- Prior 12 Months Sales
- 12-24 Month Sales
- 24-N Month Sales

Customer and Sales files are indexed by account for joins.

To further improve response for specific marketing channels, the above organization was repeated for each of 6 'product lines'. The primary product line comprised nearly 80% of the data. Differentiating the data by product line provided nearly instant response for the smaller channels.

Sample files (5%) were also created mirroring this organization. The sample tables provide response to most queries in a matter of minutes, while queries on the complete data tables requires from 30 minutes to several hours. As noted above, mainframe response to sample queries, with data stored on tape, required up to three hours. Data derived from the complete production files would generally be complete by the following morning.

This organization results in 48+ data sets. Data views are used to facilitate joins among so many tables, and separate directories maintained to differentiate sample tables from non-sample tables, Current and Non-Current Customer Tables.

'TAPE READ' UNIX SYNTAX & PERFORMANCE

The Tape loading command was the following (very simple in retrospect) Unix command sequence:

```
$ tapeutil -f /dev/rmt0 mount 1
$ dd if=/dev/rmt0 bs=128k fskip=1
of=/raw/anncraw/tapetst.xpt
$ tapeutil -f /dev/rmt0 unmount 1
```

This results in one file per tape. Each tape required approximately a minute to mount and unmount, and four minutes to read. Later use of the autoloader from a script proved the tape transfer was fast, cheap, easy and reliable.

DATA CONVERSION PERFORMANCE

The filename command was used to concatenate the tape source files. Through scripting, John Sims was able to list the directory of tapes read and 'build' the appropriate filename statement for dynamic inclusion in the final 'production' code. Syntax (two files) is as follows:

```
filename mvstape1
('/raw/data_1.raw','/raw/data_2.raw')
 recl=32260 blksize=32760
 recfm=s370vb;
```

Several initial transfers were performed to exercise the system and verify data read from tape. These transfers were monitored and it was noted that the CPU(s) were in 'Wait' state for over 80% of the time. This was as expected: most - if any - performance tuning would involve getting the system in balance, with little wait time for either disc or CPU resources.

Several packed decimal fields were noted to include non-numeric data, such as blanks. Blanks cannot be 'read' by the standard SAS S370FPD informat. John Sims provided an elegant solution via a SAS informat statement: '4040'x=0 other=S370FPD#. It was further noted that several fields were 'multi-purpose': text or numeric or even packed decimal, according to bits set in other field(s). This was handled 'in line' by testing the relevant field first via an 'IF' statement. Leigh added that customized 'C' routines could also be applied if CPU performance became an issue. Clearly the flexibility of SAS in cleanly interpreting COBOL generated production data on different hardware platforms was impressive.

Throughput, testing was both conclusive and highly successful. As of late Tuesday evening (November 10) SAS data sets had been created from the complete Master File inventory of 19 tapes in four hours (4:04). As shown below, the total time to load data from tape and create the final datasets was less than 7 hours. Actual performance far exceeded both Expected and Budgeted times.

Time in hours	**Actual**	**Expected**	**Budgeted**
Load tape data	2:14	5:00	6:00
Create Datasets	4:04	6:00	10:00
Total	**6:18**	11:00	16:00

Data set creation times were perfectly linear from 1,000,000 through 18,000,000 records. No evidence of hardware or software bottlenecks was demonstrated.

The data was read from standard, 'off the shelf' MVS Master File tapes, containing 17.6MM records of historical data. The proven ability to read these tapes at very high data rates (20GB per hour) obviates the need to contend with current MVS system resources as well as the limited available bandwidth for transferring large datasets from MVS to alternative platforms. Adding UNIX scripting to perform sequential loading of multiple tapes would further facilitate loading of tape based data, requiring minimal operator intervention.

The use of multiple files and conversion to SAS format resulted in a 36% increase in total file space required over the basic input flat file format. This was higher than expected and was eventually compensated for through adjustment of numeric storage allocation (bytes per field, see appendix).

While the present throughput rate of 25GB per hour more than amply meets the requirements for data refresh over the weekend, performance improvements may further be available through parallel processing of the Master File tapes, increasing the number of processors, and adding PCI bus channels, to further spread the reads and writes across available pipes.

EPILOGUE

As throughput rates exceeded maximum expected rates, the resulting 'excess' capacity has been used to advantage in current applications, while providing a comfortable amount of room for anticipated usage growth. It has also provided for an unanticipated but very welcome expanded functionality as a database webserver, hosting SAS/IntrNet™ and SAS/webEIS™.

ACKNOWLEDGEMENTS

The vendors involved in this project – SAS Institute and IBM - were very patient and supportive of the more cautious, risk reduction approach taken. In fact, their support determined the eventual success. The author thanks the following Institute members for their contributions: Mark Barry for acquiring and coordinating the many SAS resources applied to the project as well as the critical IBM connection, Leigh Ihnen, John Sims and Mike Chiarelli for their technical contributions to a successful benchmark, Margaret Cravar for providing the initial technical guidance on platform, and Al Schwarz , Sam Gironda, Cavanaugh Webb of IBM, and particularly Don Ferndelli of the IBM Benchmark Center.

For a much more thorough discussion of performance tuning, the reader is referred to the 'Partners' paper "*Peace between SAS Users & Solaris/Unix System Administrators*", (www.sas.com/partners/sun/performance), prepared June 1999, by Maureen Chew (Sun Microsystems), Leigh Ihnen (SAS Institute) and Tom Keefer (Sun Microsystems).

APPENDIX:

Tape storage and Backup Tape System: Part of the confusion which created such uncertainty over reading 3490 tape comes from the variety of written formats possible. It was determined that, much like cassette tape, tapes are available in two sizes, according to density. In addition, tape drives offer a hardware implemented compression feature, probably the result of improvements in hardware over time. Finally, software compression is available, based on the common Lempel-Zev compression approach. The tapes themselves of course do not come with a clear label indicating density or level of hardware compression applied. Hence the confusion.

As things turned out, the drives themselves can recognize low vs high density and compression vs no compression (hardware). It was much ado about nothing.

Prior experience had shown the need for a fast tape system for backing up the database. Maintaining an aggressive development schedule can be frustrated by long backup delays induced through slow – or worse – manual tape backup support. While the data rate supported by the 3490 was impressive, and use of UNIX compression later showed 75% reduction in space, a 2GB capacity tape would accept no more than 8GB of real data. More than 10 tapes would be required.

The 3590 media offered 10 times the storage capacity for a reasonable increase in cost compared to the time savings and maintenance free performance. Nearly 80GB of data could be compressed and piped to a single tape, one of ten available. This created an overall storage capacity of 200 GB, the equivalent of 800GB of compressed data! This would be more than adequate to contain raw data as well as user files. A side benefit, though not the best approach for reliability, the media could remain mounted on the drive, with no need for on going system backup support (and corresponding support expense). (Final Note: This simplicity was later dropped in favor of the more efficient automatic migration process offered by ADSM.)

System Parameters
System Setup parameters were:
UNIX: 3 primary file systems were established for work, source data, and SAS datasets, all striped (64K blocks). Unlimited resource availability was set including files, file space and job run times.

Base SAS System:
The config file was set up with the following options:

```
-largefile sasvlfs  (2GB+ file sizes)
-fullstimer     (performance evaluation)
-unbuflog     (reduce log buffering)
-noovp   (no overprint)

Options set at runtime:
options bufsize=64k;
```

SAS Informats were implemented through a format statement, as follows, to account for 'blanks' (hex '40') in empty fields (instead of numeric zero).

```
proc format;
  invalue ffxx
   '5C5C'x=.
   '4040'x=.
    other=[s370ff2.];

  invalue pdxx
    '4040'x=.
    other=[s370fpd2.];
```

To reduce the size of final data sets, most of the numeric data was stored in 3, 4 or 5 byte fields. The criteria for determining this differs by machine. For this RS6000 system, three bytes are the minimum storage size, based on an 11-bit exponent and 13-bit mantissa. The minimum 3 bytes allows storage to just beyond 8,000 for example. This was used to advantage for all numeric fields, including date: Three bytes could be used for numbers less than 8,192, and 7 digit Julian date stored in 4 byte numeric will be acceptable to the year 2097.

UNIX Scripts
Unix Scripts were built to perform the following functions:

```
Load multiple tapes from 10 bays, loop to
request more

Process source data into SAS data sets,
including indexing

Process peripheral data sets, reports

Backup completed datasets
```

CONTACT INFORMATION
Jeff LeSueur
BMG Direct, 31st Floor
1540 Broadway
New York, New York 10036
(212) 930 4497
Jeff.LeSueur@BMGDIRECT.com