A GPSS MODEL OF A M.V.S. SYSTEM

Ted Pollak
Ætna Life & Casualty

ABSTRACT

As part of our constant scrutiny of new technology, and its feasibility at Ætna, our computer systems simulation unit decided to investigate the IBM MP 370/168. It soon became clear to us that in order to do a credible job, many MVS features will have to be incorporated in the proposed model. Approximately six months work was needed to develop the model that the paper will be about.

The model simulates the following featues of a MVS system:

FEATURES	CONCEPTS REPRESENTED
CPU	UP or MP capability, with a black box for shoulder tapping and locks' overhead.
I/O Configuration	Channels, I/O devices, configuration flexibility, alternate channel path, logical channels.
PAGING	Check for page fault, page stealing, page out activity, flexible working set size.
Installation Performance Specifications	Flexibility to specify any IPS, as in the real system.
System Resource Manager	I/O load balancing, CPU load balancing, Workload Mgmt, Swapp recommendation. Variable nr. of address spaces in system.
Swapping Activity	Housekeeping activity involved in suspending or restarting activity for a swapped address space, as well as I/O involved in swapping it in or out.

The jobstream simulated were described probabilistically, but for validation purposes a specific benchmark jobstream was modeled. Since we had no access to a MP 370/168 system, the model was validated only relative to a UP system. Data for validation was from SMF and MF 1.

The uses of the model include modifying I/O device configurations, evaluating MP 370/168, determining cost/savings trade-offs, and evaluating different IPS's and systems timing factors for a given system and given jobstream.

The paper will deal with the methodology used to simulate the hardware configuration, and the various software features mentioned above.

INTRODUCTION

I will have to assume that the reader is quite familiar with GPSS V (Bibliography No. 1) and is reasonably familiar with the major MVS features.

The model, as it was designed, has the capability to simulate a MP 370/168 system with 193 I/O devices, tapes and disks in any mix, and main storage to the highest limits (16MB). The time unit for the model is 1 millisecond. This implies that discrete events of not less than 1 millisecond can be simulated, but events of smaller duration may be lumped together and their aggregate effect upon the system can be evaluated. The model does not simulate auxiliary storage status, or rather it assumes that we have unlimited auxiliary storage. This assumption while not theoretically true, is still a workable assumption given the large disk storage capacity of the Ætna systems. Main storage occupancy is being kept track of in 4K page frame units, i.e., unit of MAIN is equal to 4K.

JOBSTREAM DEFINITIONS

The major elements of traffic in the model are job steps and data sets, each kind being represented by a corresponding GPSS transaction. For design and study purposes a probabilistic jobstream was assumed. The calibration jobstream was different, and it will be described later.

The probabilistic jobstream had it's characteristics extracted from ACCOUNTPAK (Ætna's accounting package). Separately for testing or production, we produced cumulative distribution functions for the following measurements: CPU time/job step, nr. of disk data sets/step, nr. of tape data sets/step, channel time/disk data set, channel time/tape data set, cards read and lines printed.

HARDWARE CONFIGURATION

For a MVS model it is preferable to have a flexible configuration, able to simulate UP or MP. A matrix should be defined with as many rows as the maximum number of I/O devices expected to be attached. Column nr. 1 will carry the device number, column nr. 2 will carry the primary channel number, column 3, 4, 5 carry the three alternate channel numbers to be available if MP is configured, and column nr. 6 carries the logical channel number. If a symetric system (for MP) is configured, fine. However, if no alternate path exists for a set of I/O devices, for this set initial the columns 3, 4, 5 with the same primary channel number as column nr. 2.

Before attempting to spend I/O time at a certain device, the status of the device and the various channel paths to it are examined via Boolean variables testing for FNU (facility not in use). If no free path exists the data set transaction will be linked to the chain indicated by the logical channel nr. As users of this logical channel end a loop, they will unlink all transactions linked to their logical channel for a new try.

The flowchart on Figure Nr. 1 gives the succession of events in the life of an active address space.

Note 1. Cards, lines, number of data sets, I/O and CPU time generated by user.
Note 2. Start data sets in parallel to CPU activity.
Note 3. Paging consideration.
Note 4. A decision will be made which CPU will execute this time slice.
Note 5. Alternate path and should tapping will be simulated.

Jobs are initiated on a FIFO basis. The initiator number (stored in PF9) becomes a key for many things. It identifies all the data sets belonging to the step, the paging chain number to store page transactions, (to be explained in the description of the "paging routine"), the group number to store page ID numbers (see descr. of "paging routine"), accumulator matrices and save values for service units absorbed by the transaction and logic switches that will transmit swap recommendations. All this is accomplished via an EQU statement.

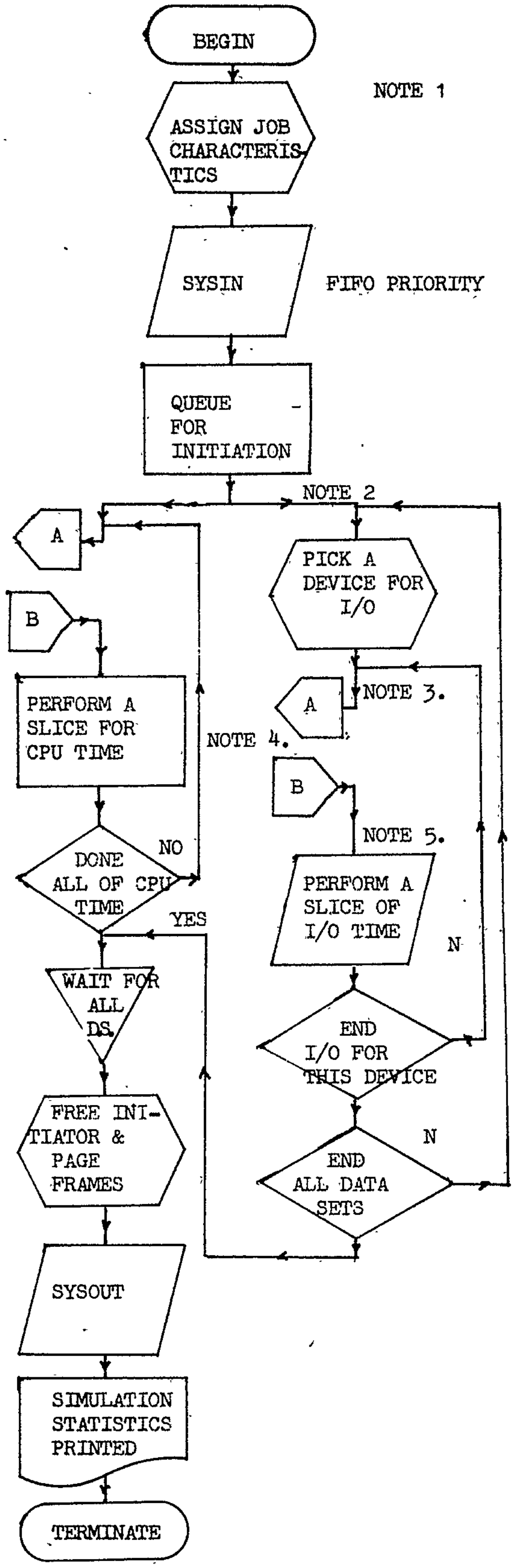
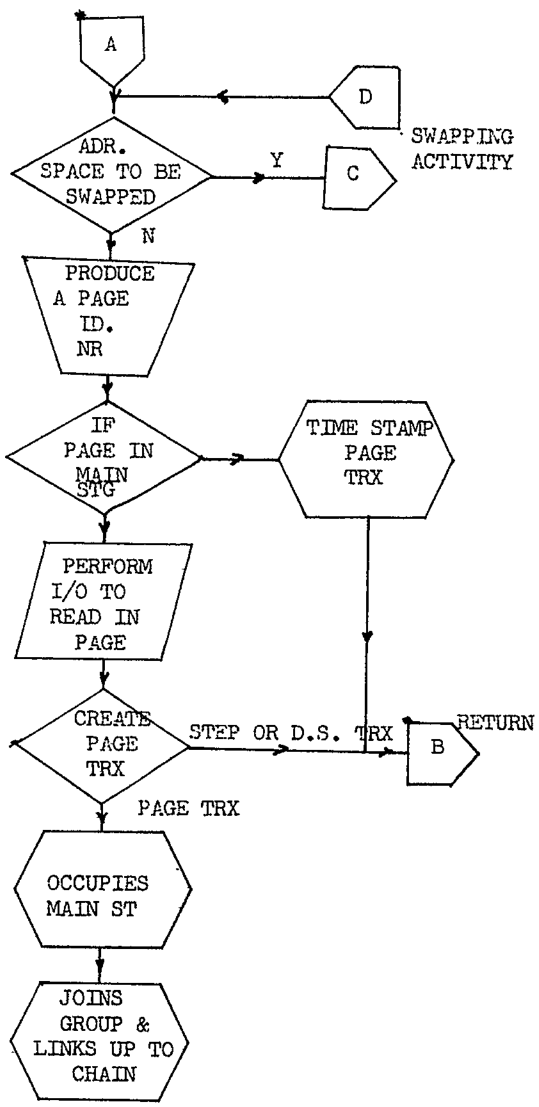


Figure 1.

Once initiated, the step and data transactions loop reducing the amount of CPU and I/O time allocated for them. At each looping, a branch is taken to the "paging routine". When the transaction returns, accumulators for service units absorbed are updated. These accumulators will be discussed in the description of the "Workload Manager", routine. When the times allocated for CPU and I/O respectively are spent, the transactions will ASSEMBLE and free initiators and page frames occupied, and the step will terminate.

PAGING ROUTINE

Step and data set transactions branch to this routine at every loop. The swapping logic switch is examined to test if swap out is ordered. If swap-out is ordered, a branch to the "swap-out routine" is taken. Else, the transaction proceeds with the paging activity.



PAGING CONSIDERATIONS

Figure 2.

Paging is simulated in the following fashion. Each executing program is associated with a region size (i.e., 300K), corresponding to the region size it would need in OS/MVT. This region corresponds to a number of pages (i.e., 300/4 = 75 pages), namely p1, p2, p3 ... p75. Page ID numbers are generated randomly with the following technique:

```
ASSIGN   1, V$ADSPC, PF INDICATE WORKING SET
         SIZE

ASSIGN  10, V$PGID, PF PRODUCE A PAGE ID NO.

ADSPC   FVARIABLE   PF1/4*FN$WRKST

PGID    VARIABLE    RN4@PF1+1

WRKST   FUNCTION    RN2, C2  WORKING SET % OF
                    ADDR. SPACE

.90, .20/1.0, 1.0
```

where PF1 contains the size of the address space in K's.

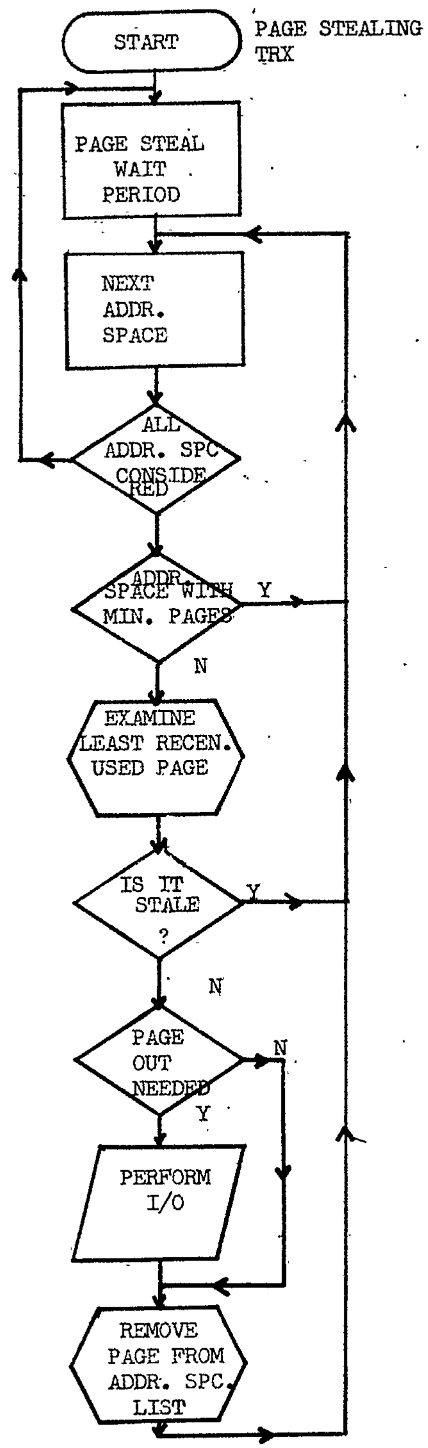
A group with number entries for each address space will keep track of pages in main storage. As pages ID numbers are generated, they join the group. When page requests arrive, this group is examined to see if pages are in it. If the page number requested is not in the group then a page fault is simulated.

A PAGING chain with FIFO discipline will exist for each address space whose transaction members will contain the job and page information of pages in main storage. Every time a page fault occurs a transaction is created that will be linked to the PAGING chain. If the requested page was already in main storage we will UNLINK and re-LINK the corresponding transaction to the PAGING chain. When a page out is desired, we UNLINK the first transaction (least recently used) of the chain, remove the appropriate page number from the group, if necessary execute the I/O for page-out (if alteration to this page occurred) and terminate the transaction. This discipline will page out the least recently used page.

When a SWAP out or a job termination occurs all the pages associated with the job number in question will be handled as if a page-out occurred.

Locality of reference will also be considered. Locality of reference implies that during a program's exeuction, it's reference pattern will dwell within a relatively small number of pages (compared to the total in a program) for relatively long time periods. This locality of reference varies, obviously, from program to program. To simulate good locality of reference, instead of the 75 pages (for our example) only a fraction of these will be generated. Thus, the same phenomenon (reduced page faults) will occur in the model as in the real system. The effect of less pages generated will be less page faults.

The multiplier to reduce or increase page numbers generated simulating good or bad locality of reference will be tuned to real systems statistics. This is done by manipulating FN$WRKST. Real systems statistics will be used for page-outs and swap-outs.



PAGE STEALING ROUTINE

FIGURE 3.

Each job executing will have his own PAGING chain, corresponding to the address space. A "page stealing" routine will examine each group at regular time intervals (TA). The groups that contain pages below a threshold (Pa) will be ignored. Those that are above it will have their pages examined for the last time it was referenced. If the time since it was referenced was longer than a maximum value (tB) the page will have it's page frame stolen.

The system must have a threshold for available pages (Ps). When the available pages go below this threshold whole address spaces are swapped out, instead of just pages being stolen. If this is not enough to reach the limit (Ps), page stealing will continue.

tA and tB will be set dynamically
Pa will vary between 5-20 in increments of 1
Ps will vary with a lower limit of 10 page frames

These constants are dynamically adjusted. If CPU utilization is higher than 95%, the constants will be dynamically adjusted so that less page stealing will be done. To do this tA, tB and Pa and Ps will be increased. When CPU utilization falls below 80% all the above constants will be increased. The range for tA and tB will be determined experimentally.

Systems Resource Manager

This routine monitors the CPU, I/O and the system's workload. References 2 and 3 describe the algorithms used to compute recommended values for swapping.

The CPU load balancer ranks address spaces by their use of CPU cycles. The recommended value considers the CPU utilization during the analysis period as well as the CPU utilization caused by the given address space. The intermediate CPU utilization is computed from the facility utilization statistics carried by GPSS. The magnitude of the CPU recommended value is given by the following formula.

$$RV\ (CPU) = (D^2R)$$

Where D is the distance the CPU utilization is from acceptable range, and R is the CPU rate of the address space. For a swap candidate, the RV is positive for an under utilized CPU and negative for an over utilized CPU.

During the I/O loopings counts are kept for the number of EXCP's executed for each logical channel. The I/O recommended value is computed in the normal positive/negative number by the following formula:

$$(I/O\ RV) = D^2R$$

Where R is the EXCP rate on the most imbalanced logical channel and D is the distance from the acceptable range of delayed channel requests, in percentage points. For example, for UP, 70% is the high threshold value. If 84% of the total requests on that logical channel are delayed, then D = 84-70 = 14.

WORKLOAD MANAGER

Associate a save value (accumulator) with each address space, and a save value for the system workload level.

Establish a matrix for performance group information. The row number corresponds to the performance group number. Columns 1, 4, 7 (3* n + 1) carry the length of the period that the transaction (job or jobstep) is associated with a performance objective. Columns 2, 5, 8 (3* n +2) carry the performance objective during the period indicated in the previous column. Columns 3, 6, 9 (3* n) carry the interval service values which must be accumulated before a swap is considered. (See end of description).

Performance objective functions will map transaction service rates to normalized workload levels. Thus given a performance group and a service rate the normalized workload level is obtained via the MPAS and OBJ 1, 2, 3, ... functions. (See end of description).

Transactions will be assigned a performance group soon after they enter the system. As the transactions use CPU and I/O time the save value corresponding to their address sapce will accumulate the service units used during the respective activity. This is done using the proper formula and the Installation Performance Specifications (IPS).

The System Resource manager routine will be activated at certain time intervals, and will examine each address space. From the accumulated service units it will determine the service rate the transactions absorbed during the last examination interval. Via the matrix, the routine will establish the performance objective the transaction belongs to, and via the MAPS function will obtain the "transaction normalized workload level". If this "transaction" normalized workload level is greater than the "system" normalized workload level, then the transaction is "behind". Otherwise the transaction is "ahead". If a transaction is "behind", it means that it has not absorbed enough of the service units planned for it. Therefore it's swapping recommendation will be reduced, so that it should not be swapped out readily. The reverse procedure is applied to transactions that are "ahead" in their absorption of service units. Their swapping recommendation will be increased.

Once all address sapces are examined a "system" normalized workload level is computed, as the average of all "transaction" normalized workload levels.

Matrix for Performance Group Information

	COL 1	COL 2	COL 3	COL 4	COL 5	COL 6....
GRP1	700 serv.u Period 1	OBJ2	500 serv.u	800 serv.u Period 2	OBJ3	600 serv.u
GRP2	800 serv.u Period 1	OBJ1	600 serv.u	700 serv.u Period 2	OBJ2	500 serv.u
GRP3	600 serv.u Period 1	OBJ3	500 serv.u	900 serv.u Period 2	OBJ.1	800 serv.u

FN$MAPS leads TRX to these functions.

OBJ1 TRX service rate 0.......400....900
workload level 100 5 0

OBJ2 Transaction service 0...300....900
workload level 100 5 0

OBJ3 Transaction service rate 0...600....900
workload level 100 5 0

The System Resource manager will compute the swapping recommended value for each address space as the algebraic sum of the RV's of the CPU, I/O load balancer and the Workload manager. If the global RV is bigger than a swapping threshold, the logic switch for the given address space will be set. When this address space will branch to the "paging routine" it will be swapped out. If the RV is less than a swap in threshold, a swap-in will proceed by UNLINK-ing the address space (step trx) in question, and sending it to the SWAP-IN sub-routine.

SWAPPING ACTIVITY HOUSEKEEPING

The following will describe the housekeeping activity that is to take place once a decision is made to swap in or two swap out a given address space.

A chain (JOBS) is established to store the step and data set transactions while they are swapped out. Another chain (SWAP) will store the page transactions. Each address space has a logic switch associated with it. When a step or data set transaction is branching to the PAGER routine (to verify if the page to be processed is in main storage), it will first check the status of the logic switch. If the switch has been set (by the System Resource Manager routines), the transaction will be swapped out (by branching to the swap routine).

"SWAP-OUT LOGIC"

The data set transactions will be LINK-ed to the JOBS chain without any change. An I/O activity will be performed to read the pages of the working set to auxiliary storage. The step transaction

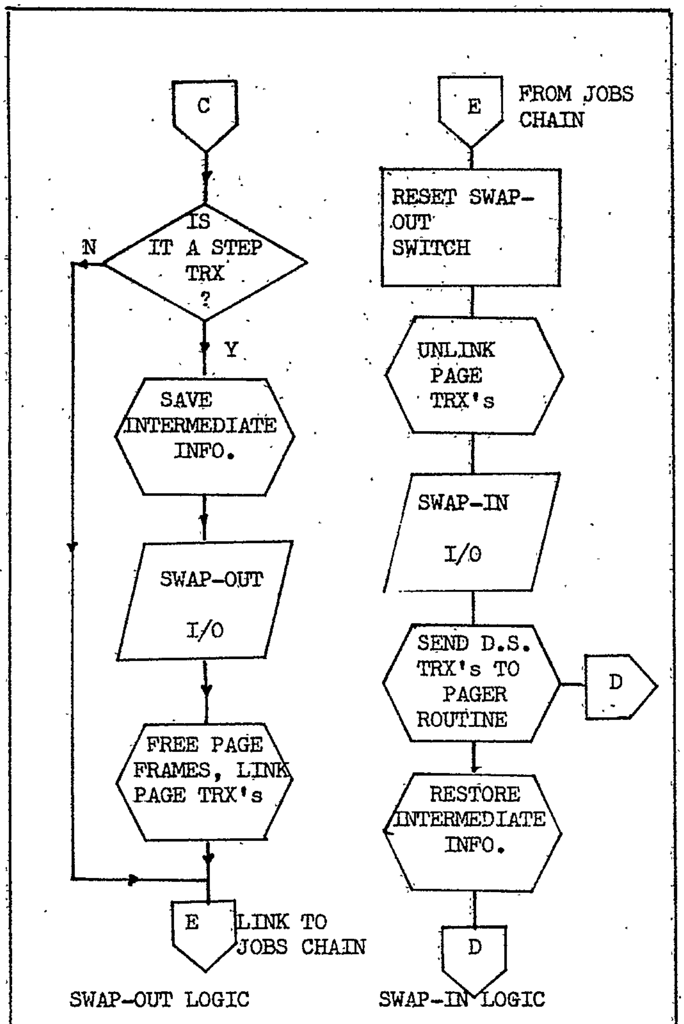


Figure 4

will UNLINK all page transactions from the address space chain, free their frames and LINK them to the SWAP chain. The step transaction will then be time stamped and LINK itself to the JOBS chain.

"SWAP-IN LOGIC"

When the Systems Resource Manager routine determines that a job should be swapped in, the job's step transaction will be UNLINK-ed off the JOBS chain. The logic switch will be reset, indicating a no-swapout condition. The page transactions will be UNLINKed off the SWAP chain and will occupy their allocated page frames. An I/O activity will be performed to read in these pages. When all pages have occupied their frames the data set transactions will be freed and sent to the PAGER routine. Lastly, the step transaction will start its own regular processing.

MODEL VALIDATION

At the time of this writing, Ætna Life and Casualty does not have an installed MP370 system. Therefore, we calibrated against a UP system running under MVS.

The Operating Systems Research unit at Ætna devised a benchmark jobstream of 8 distinct jobs whose profile is a representative of the kind of jobs being processed in this data center. An operating system is tuned and evaluated by running these 8 jobs over and over, and measuring systems performance. This benchmark jobstream was used to calibrate the MVS model. Data from SMF and MF1 was obtained from running the benchmark jobstream.

This data was incorporated in the model as a matrix for each job that was initiated. Each row corresponded to a job step. Column No. 1 carried step CPU time, Column No. 2 carried step region size, Col. No. 3 had the number of data sets for this step, Col. No. 4, 5, 6 etc., carried the number of EXCP's for each data set.

100 transactions were started. Each had a job number modulo 8. The jobs picked up their step and data set information from the matrices corresponding to their job and step. The model was stopped after executing 96 jobs. The calibration was accomplished by tuning the various constants until the model statistics corredponded to the systems statistics. These constants were:

- Page list distribution function FN$WRKST
- Size of the minimum working set
- Fraction of stolen pages to be paged out
- Size of the "page stealing routine" interval (TA)

The results of the calibration effort were:

MEASUREMENT	MODEL STATISTICS	SYSTEMS STATISTICS	DIFF. %
CPU Util	92.7%	94.3%	1.7
Page in Rate (page/sec)	.18	.18	0
Page out Rate (page/sec)	.56	.64	12.5
Swap Rate (page/sec)	.74	.75	1.3
Service Unts. Accumulated	2,236,868	2,168,712	3.1
Elapsed Time	1435 sec.	1523 sec.	5.8

CONCLUSION

Initial studies, using a simpler form of the MP model indicated that a MP system would not be cost justified at our company, as things stand now. However, the development of a MVS model led to further activities. A new project was started, to develop a front-end to the MVS model so that it could accept preformated job accounting data as input. We plan to use the model to evaluate different Installation Performance Specifications with realistic input data, once a MVS system will be up and running at Ætna.

I want to thank the following people for their aid in this effort: Martha C. Kalar for working and helping me throughout the project, Jeff Alperin and Mike Cuddigan for their explanations of the innards of MVS, Erich Aust and Dick Stoltz for guiding me along the way, and Debbie Jordan for preparing the manuscript.

BIBLIOGRAPHY

1. GENERAL PURPOSE SIMULATION SYSTEM V. User's Manual SH20-0851-1 I.B.M. 1973

2. OS/VS2 SYSTEM PROGRAMMING LIBRARY; Initialization and Tuning Guide (VS2 Release 3) GC28-0681-1, I.B.M. 1975

3. I.B.M. Systems Journal; OS/VS2 - System Resource Manager Lynch and Page, No. 4, 1974

4. Introduction to Virtual Storage in System 370; Student Text I.B.M., 1972.