AN ANALYSIS OF IMS THROUGH SIMULATION

by

James W. Chrissis

Thesis submitted to the Graduate Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

Industrial Engineering and Operations Research

APPROVED:

J. William Schmidt

Vinod Chachra

Robert P. Davis

October 1977

Blacksburg, Virginia

ACKNOWLEDGMENTS

I would like to express my appreciation to the many people who have contributed to the completion of this thesis. In particular, to my advisor, Dr. J. William Schmidt, for his assistance and guidance throughout the research. Also to Dr. Vinod Chachra for his assistance with the IMS side of the research and to Dr. Robert P. Davis for his encouragement.

I am also grateful for the cooperation and assistance of the Systems Development Department and the Virginia Tech Computing Center, especially . Special thanks are also in order for for her understanding and confidence and for her typing efforts. Finally, I would like to thank my parents, to whom I am especially grateful.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Area of Investigation

Computerized Data Base Management Systems (DBMS) are in widespread use, with continued growth anticipated for the foreseeable future. The volume of data maintained and manipulated and the number of users desiring access to this data make these systems essential to the effective operation of many organizations. A number of factors affect the performance of a DBMS including the volume of data, number of users, data base design, program efficiency and system configuration. On-line systems have been developed to give the user almost immediate access to the data within the system, but poor system performance can defeat this objective. For this reason, performance studies of DBMS are often necessary when system response deteriorates to a level below that considered to be the minimum acceptable, with the intended purpose being relief of the problem. Studies of this type are also useful in the identification of solutions to problems anticipated in the future. The research presented herein is a performance study and analysis of a specific DBMS. Although the results apply to the specific DBMS studied, the methodology utilized may be applied to any information system of a similar structure.

The DBMS under investigation is IBM's Information Management System/Virtual System (IMS/VS). IBM represents a majority share of the computing systems market and IMS* is the DBMS in use at many installations. A number of IMS variations are available, each designed to meet the specific needs of the particular installation. The Virginia Polytechnic Institute and State University IMS/VS is used as the source for investigation. Specifically, the operation of the on-line IMS facility is analyzed.

1.2 Introduction to IMS

In this section some basic concepts of IMS are presented. More detailed descriptions and additional information on IMS can be found in References [3], [4], and [6].

IMS is IBM's principal entry in the DBMS market and provides the functional capability to store, access and manipulate large volumes of data. It is not an independent entity; it requires a host environment in which to operate. The user must provide the operational environment and design and implement the application programs to meet specific requirements.

IMS/VS extends the capabilities of the Operating System for Virtual Storage (OS/VS) to the Data Base (DB) and Data Communication (DC) environment. The basic system, the DB system, provides facilities for defining, creating and maintaining IMS/VS <u>Data Bases</u>† and for executing

---

* Hereafter, IMS/VS may be referred to simply as IMS.

† Newly introduced terms may be underlined in some cases. More specific definitions of these terms can be found in the glossary.

application programs in the batch mode. The data communication feature provides for the transmission of messages between IMS/VS terminals, both local and remote, for on-line processing.

Figure 1.1 shows the components that constitute an IMS/VS DB/DC system and the relationship to the OS/VS control program. Each of these components is discussed below.

Control Facility

The initiation and control of the various IMS/VS facilities (functional units) is provided by the IMS/VS control facility. The control facility performs the following functions:

1. Loads control blocks from control block libraries that define user programs and the data bases to be accessed.
2. Loads action modules necessary for program execution.
3. Loads the user application program into the region and transfers control to it. It should be noted that each message processing program executing concurrently operates in a unique region or partition of the CPU.
4. Provides for a function designated program isolation. This provides the ability for two or more user programs to use a data base concurrently.
5. Executes the system logging functions necessary for proper restart or recovery of the system.

Data Base Facility

The data base processing capabilities of IMS/VS are provided by a facility called Data Language/I (DL/I). The functions supported by DL/I

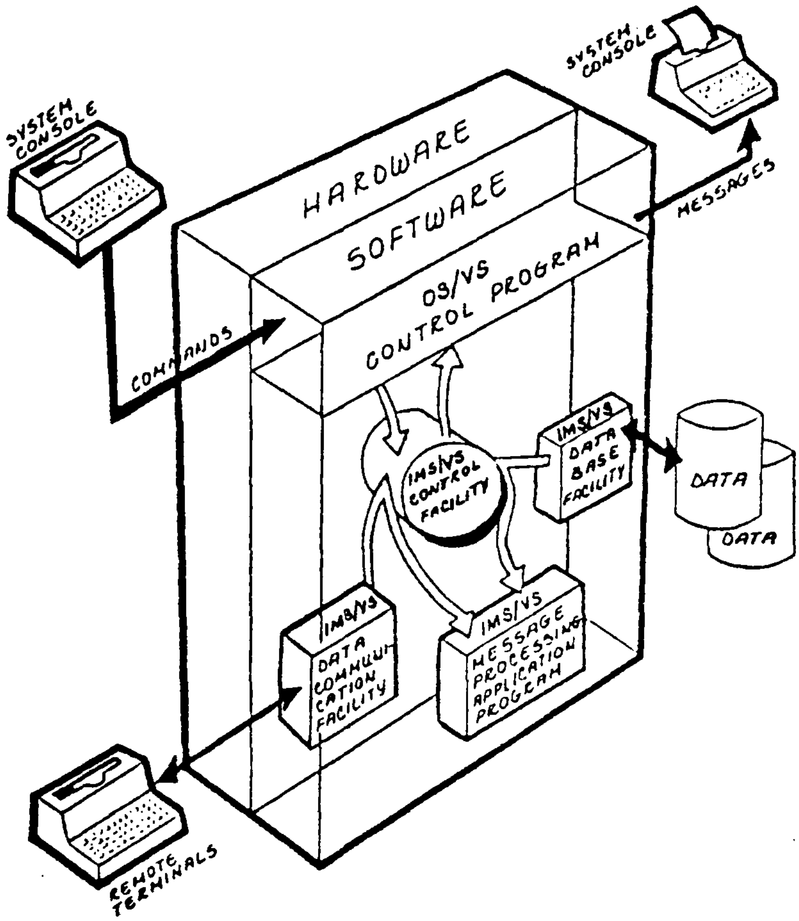


Figure 1.1 Components of an IMS/VS DB/DC System.
(Reference 4)

are definition, creation, access, update, reorganization and recovery of the data bases.

Data Communication Facility

The data communication facility supports a wide variety of remote and/or local terminals and devices. The terminals provide the user with a means of timely access to the data base. IMS/VS can send a variety of message types for multiple applications.

Figure 1.2 is an expansion of Figure 1.1 showing the OS/VS regions in the multiprogramming environment. Separate OS regions or partitions are used for message processing. Each region or partition may have unique characteristics such as size, priority and classes of messages to be processed in the region. These characteristics define the class of the processing region or partition. The message scheduling facilities of IMS/VS initiate message processing program load and execution through supervisor call routines added to the OS nucleus.

The IMS/VS input message scheduling algorithm is controlled by the user. Four parameters must be provided by the user at the time each message type is defined:

1. Normal priority - the priority at which messages of this type are normally processed.
2. Limit count - when the count of messages of this type in the input queue is greater than or equal to the limit count, the normal priority is raised to the limit priority.
3. Limit priority - when the limit count equals the queue count (number of messages of this type in the input queue), the

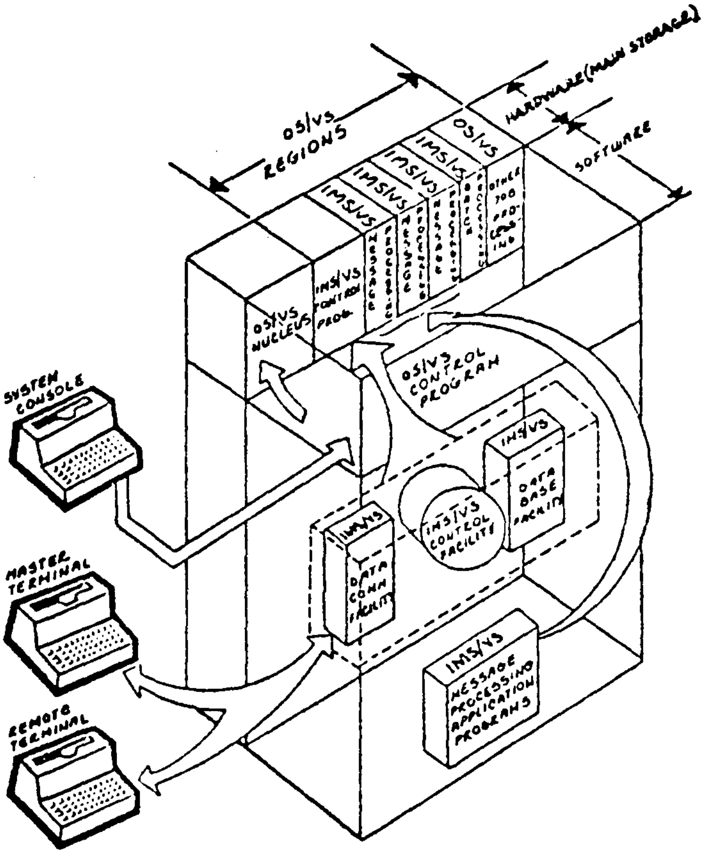


Figure 1.2 IMS/VS and the OS/VS Regions.
(Reference 4)

normal priority is raised to the limit priority until the queue count returns to zero.

4. Message class - determines the region of partition into which the message type will be scheduled.

An example will help illustrate this. Consider a message with the following:

Normal Priority = 2

Limit Count = 10

Limit Priority = 15

When the number of messages of this type in the queue is greater than or equal to 10 (the limit count), the priority is raised from 2 (the normal priority) to 15 (the limit priority) until the number of this type awaiting processing becomes zero.

1.3 System Description

The system to be analyzed in this study is composed of the following components:

1. Network of administrative terminals.
2. Input queue for message arrivals.
3. CPU regions for message processing.
4. Data bases for storage of information necessary for message processing.
5. Communication channels for transmitting information from the data bases to the processing regions.

The entire message processing cycle is initiated when a *transaction code* is transmitted from a terminal in the network of administrative users to the data communication facility. Each arrival is processed by the message format service, which converts the input stream to program useable form, and is entered in the input queue to await processing.

The next message to be removed from the input queue for processing in a region is determined by a priority scheme. The priority scheme can be described as follows:

1. Messages belong to classes and have a normal priority within each class.
2. Message processing regions process all the messages within one class, then in the next class and so on.
3. Different regions can have a different priority ordering of the classes of messages they process.
4. Within each class, messages are processed in order of priority; from highest to lowest.

The priority of a message may be dynamically altered based on input queue characteristics overriding the normal priority. If the limit count for a particular message type is equalled or exceeded, the normal priority is raised to the limit priority until the queue count becomes zero. At that point the normal priority is restored.

Another situation that overrides the priority scheme occurs when a program that has been processing in a region is reuseable. If a program is reuseable and the processing limit is greater than one, the program retrieves the next message which requires that program for processing regardless of message class or priority. This process continues until either all messages requiring that program have been processed or the processing limit has been reached. The program is then removed from the region and the normal priority scheme restored.

In addition to the processing region, a message being processed may require information contained within one or more of the data bases. This information is transmitted to the processing region via a communications channel. If a channel is processing a data base call, any other calls that require that channel must wait until the channel is free. All calls, whether IMS or external, are processed first-come, first-served.

Upon completion of the processing required for a given message, the output to be transmitted to the user is placed in an output queue. Output transmission for the system studied here is virtually instantaneous and places no load on the input and processing segments.

The communications channels that are used for IMS processing may also be used for non-IMS processing. External (to IMS) processing places a load on the channels which affects IMS performance. The impact of changes in this external load on IMS performance will be analyzed in the study.

The specific system under consideration is illustrated in Figure 1.3. There are two processing regions and two data communication channels. The IMS data bases are stored on two disk packs: one pack on each of the channels.

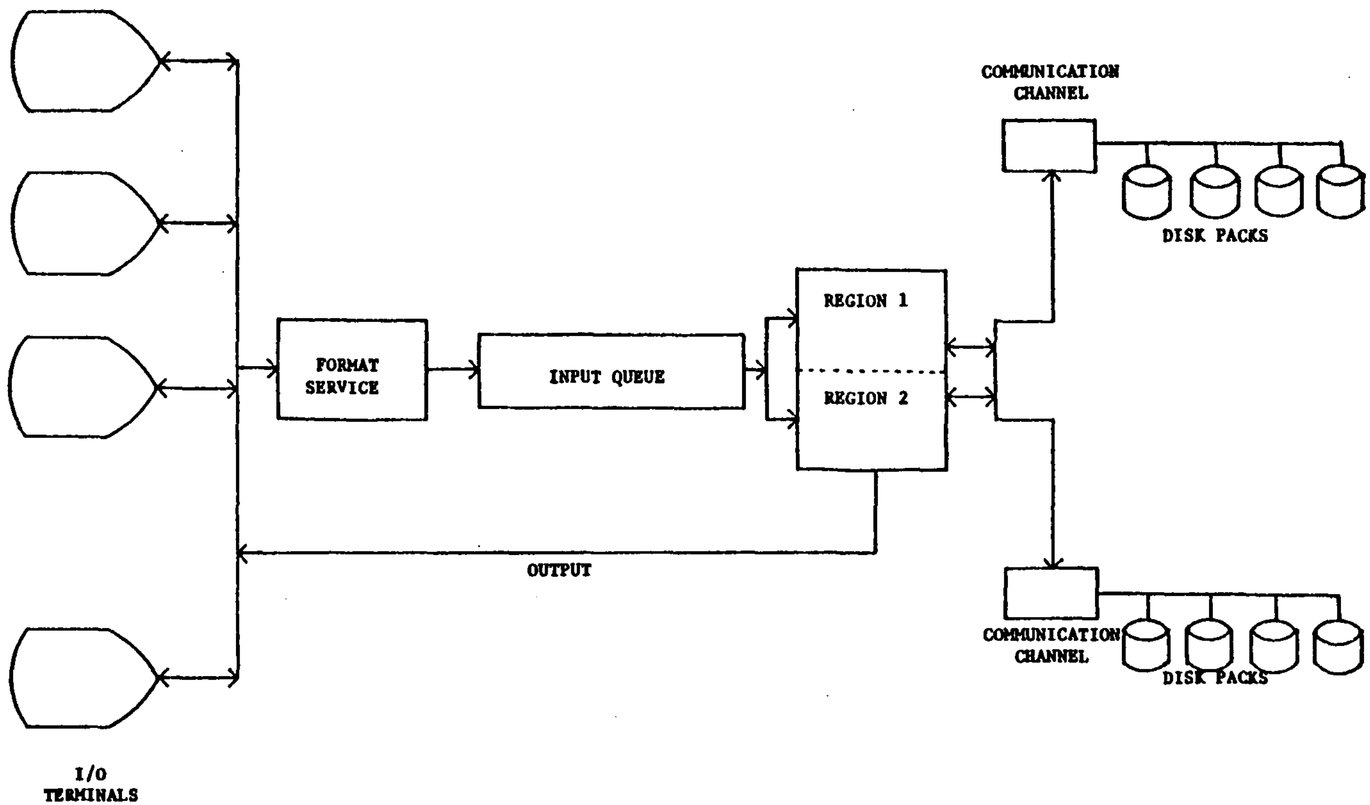


Figure 1.3 The IMS/VS Configuration to be Analyzed.

1.4 Objectives of the Research

The objectives of this study are:

1. To develop a simulation model for a typical IMS/VS facility.
2. To validate the model using data collected on the VPI&SU system.
3. The analyze the performance of the system under a variety of operating conditions using the simulation model.

The variation in operating conditions studied are as follows:

1. Increased IMS load.
2. Increased external load.
3. Addition of a third processing region.
4. Addition of a third communication channel.
5. Changing the priority class configurations.
6. Variation of the distribution of data bases to communication channels.

These will be investigated both individually and in combination. IMS response depends on both IMS and external loads. For this reason it is desireable to investigate the effect that changes in IMS load and external load will have on IMS performance. The effect that changes in priority class configuration and data base distribution have on IMS response will be investigated to determine if more efficient configurations are possible. The results of analyzing the variations will be used to determine the system configuration that yields the greatest improvement in system performance as a function of total system load.

Response time will be used as the measure of system performance. Response time is defined as the elapsed time between the submission of a message for processing and its completion and transmission to the user. In addition, other elapsed times within the system, such as region time and time required for data base calls, will be used as indicators of the performance of the various components, primarily for validation purposes.

1.5 Review of Supporting Literature

A search of the available literature has shown that little work has been done on information system optimization or simulation of data base management systems. Roach [8] discusses a simulation technique called SCIMS, which can be used to design optimal performance IMS data bases. In the discussion of modeling IMS he states that "analytical solutions do not exist [now], if in fact they could ever be developed." He also points out the difficulties involved in developing and validating a simulation model of a system as complex as IMS.

SCIMS is a validated flexible and comprehensive simulation model of IMS. It runs on any configuration that will support IMS, requiring from 10 to 60 minutes per run. Each run produces a set of detailed reports which are used for evaluation.

SCIMS considers an existing hardware configuration and improves on the data base design. Roach states that hardware configuration is considered, but does not discuss specifics. It can only be concluded that any hardware improvement that occurs is an indirect consequence of the data base analysis. The simulation model developed in the course of this research will, given a present data base design, show how changes in the system configuration affect performance. Also, the proposed simulator could be used by any installation that supports FORTRAN, whether it supports IMS or not.

Edgecomb and Thompson [2] have treated the problem of validation of simulation models for complex computer systems. They summarize the major problems encountered and suggest some possible solutions for these problems. They point out that while computer and simulator technology have improved, the validation process has had little attention and support. Simulator complexity is one problem area in the validation process. Obviously, the more complex the system, the more complex the model will be and the more difficult the validation is. Data availability is another bottleneck in the validation process. The authors point out that "system documentation was never designed to support simulation projects except at a very general level." Hardware and software monitors must be used for gathering performance data to be employed in building and validating the model.

The simulator developed herein is quite complex and difficulties were encountered in its validation. However, data from IMS monitors was available to aid in the validation phases.

Tryggestad [11] discusses simulation techniques for the evaluation of data base management systems. He highlights one critical point: "the accuracy of results depends on the accuracy of the input data and the care with which the simulation activity is conducted." He also makes a point for the use of analytical modeling techniques for application to physical level (program I/O and CPU activity) data to obtain results similar to those produced by simulation studies.

Lavenberg and Shedler [5] present a stochastic model of processor scheduling in IMS. The model represents the processor scheduling and queueing due to contention for resources. The authors state that "the model is not used in a performance study of IMS, nor is it proposed that the model developed is one upon which a performance study of IMS should be undertaken." They also point out that "the model should be viewed as illustrative of stochastic models which can be constructed to incorporate algorithms for processor scheduling." The model presented illustrates only the processor scheduling and is not necessarily intended for use in a full scale IMS model. The specific analysis of processor scheduling is beyond the scope of this research.

Schneider and Connolly [10] present a generalized data base system simulator based on the Data Independent Accessing Model I (DIAM I). The simulator may be used to conduct studies relevant to the selection and use of data base systems, including:

1. Comparison of the performance of two competing data base management systems for the same application.
2. Comparison of the performance of a particular DBMS under various applications.
3. Comparison of the performance of different host system configurations being considered.

No application of DIAM was mentioned for IMS.

DeLutis and Smith [1] describe a special purpose simulator developed to investigate the behavior of network and hierarchical classes of DBMS's. The simulator provides the facilities for characterizing the features of DBMS's. The following may be defined for a DBMS:

1. Data structure
2. Data base content
3. Data manipulation language operations
4. Task management and resource allocation
5. Interfaces to application software and the operating system
6. Data translation algorithms
7. DBMS performance measures

The authors specifically mention the applicability of the simulator to IMS from a data base design point of view. The research presented herein assumes the given data base design and attempts to improve on the system configuation.

Some work has been done in the area of computing system modeling and analysis, as opposed to an on-line data base management system, which this research is concerned with. Pollak [7] discusses a GPSS model of an MVS system and Rosenshine and Zenakis [9] have analyzed the operation of a computing center. However, neither of these analyses considers IMS.

The shortage of IMS simulation studies demonstrates the need for such an undertaking. As IMS is one of the most commonly implemented data base management systems, the study conducted in this research effort may be of general interest.

1.6 Outline of Succeeding Chapters

In Chapter II the formulation of the simulation model is presented. Included is a discussion of the collection and aggregation of input data.

The validation of the simulator and methods for analysis of output data are presented in Chapter III. In Chapter IV the results obtained from the simulation model for the various operating situations is discussed. Chapter V contains the conclusions, recommendations and extensions for future research.

CHAPTER II

APPROACH TO THE PROBLEM

2.1 Introduction

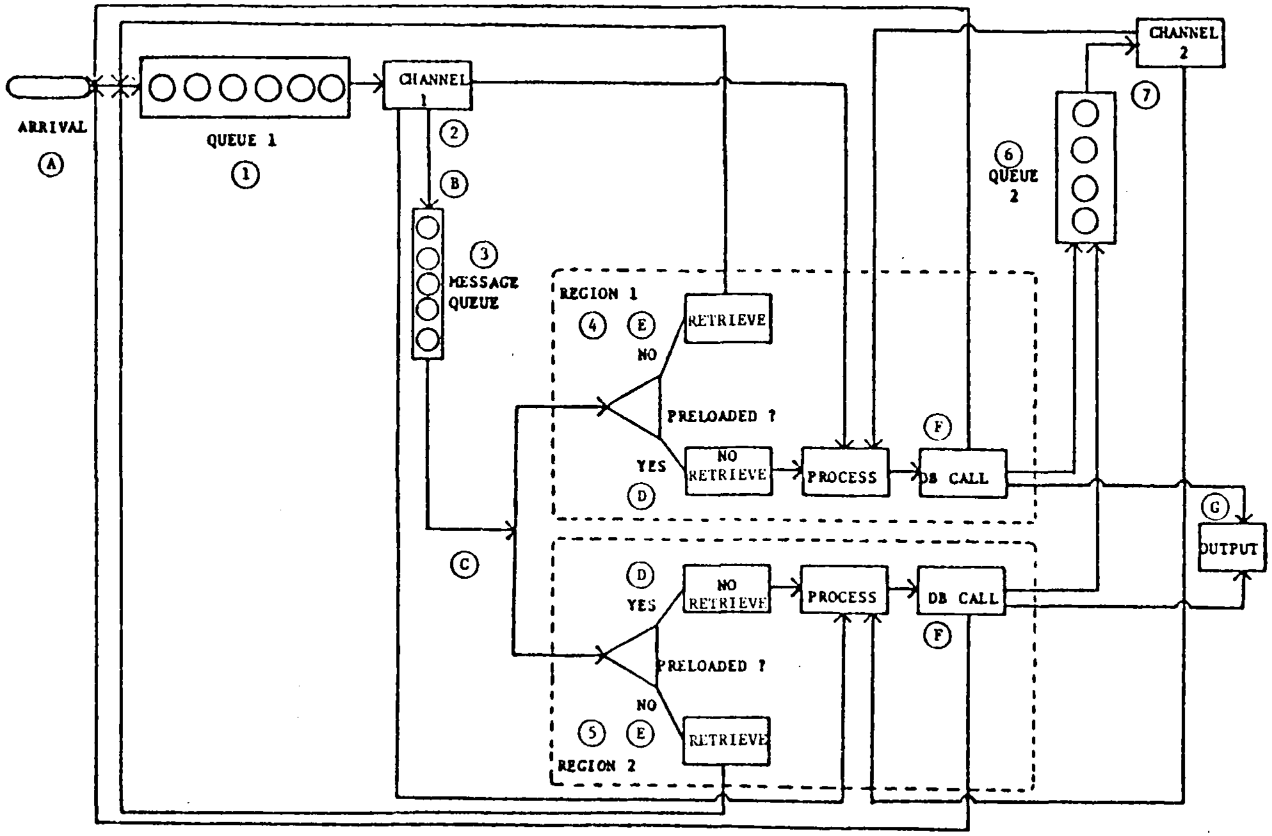
The operation of the existing system and proposed variations are to be analyzed to determine system configurations which can lead to improved system performance. Because of the cost of computer hardware and software, it would be prohibitive to experiment with different configurations using an actual system. A system model which captures the essential characteristics of the system and the effect on the dependent variable (i.e., response time) caused by changes of the independent variables (IMS load, external load) under experimental configurations is desired.

Because of the relative complexity of the system, particularly from a queueing point of view, and the variety of system configurations which must be considered, a simulation model would seem to be the most utilitarian means for evaluating system effectiveness. A simulation model would be more appropriate than a mathematical model because of the difficulties involved in modeling and analyzing a system of this complexity using available mathematical techniques.

2.2 The System Model

A diagram of the system, modeled as a series of queues is given in Figure 2.1. The model consists of the following components:

1. Queue 1 - waiting line for services at channel 1. Arrivals are processed first-come, first-served.



1 through 7 represent system components.
A through G represent points in message flow.

Figure 2.1 The System Modeled as a Series of Queues.

2. Channel 1 - communications channel for data transmission.
3. Message queue - waiting line for messages to be processed. The priority scheme was described in section 1.3.
4. Region 1 - message processing region.
5. Region 2 - message processing region.
6. Queue 2 - waiting line for service at channel 2. Arrivals are processed first-come, first-served.
7. Channel 2 - communication channel for data transmission.

2.3 Transaction Flow

Figure 2.1 can also be used to illustrate the flow of a typical arrival to the system.

A. A message arrives from one of the user terminals. All incoming messages are entered in Queue 1 (1) to await processing by the message format service, which requires channel 1 (2). These format service entries have priority over all other entries in Queue 1.

B. After processing by the message format service, the message is placed in the message queue (3) to await processing.

C. The next message for processing is selected according to the priority scheme and enters the appropriate processing region (4 or 5).

D. If the application program required to process the message is preloaded, no program retrieval or scheduling is necessary and processing begins.

E. If the application program required to process the message is not preloaded, the program must be retrieved from the program library and scheduled for processing. Program retrieval requires processing by channel 1. After the program retrieval and scheduling is completed, message processing begins.

F. During processing, information contained within one or more of the data bases may be involved. When this occurs, the program issues a call to the appropriate data base. The data is transmitted to and from the processing region via a communication channel (2 or 7 depending on which data base is involved). If the channel is busy, calls are entered in the queue (1 or 6) until the channel becomes free. These data base calls are processed first-come, first-served.

G. When message processing is completed, the output is transmitted to the user.

2.4 The Simulation Model

The simulation model is a next event simulator; simulated time is updated at the occurrence of each significant event, independent of the time elapsing between events. The simulation program is written in FORTRAN because of familiarity with the language and a degree of flexibility which is not available with one of the special purpose simulation languages.

The main program consists of the initialization of necessary parameters and the next event routine. The events that can occur are:

1. End of service on channel 1.
2. End of service on channel 2.

3. End of scheduling for the program in region 1.

4. End of scheduling for the program in region 2.

5. End of block building for region 1.

6. End of block building for region 2.

7. End of no scheduling for the program in region 1.

8. End of no scheduling for the program in region 2.

9. End of processing in region 1.

10. End of processing in region 2.

11. IMS arrival.

12. External arrival to channel 1.

13. External arrival to channel 2.

14. End of simulation.

The next event routine determines which event will occur next. The event which corresponds to the smallest event time is the next event to occur. Calls to the appropriate subroutines are made from the main program.

Subroutines are used to keep track of the transaction flow through the various stages of the system. The remainder of the simulation model consists of a number of subroutines. Basically, each subroutine performs a separate bookkeeping function, updating the progress of the transaction as it moves through the system. The subroutine identifiers and the functions each performs are:

CHNL1, CHNL2 - simulates the operation of the communication channels (channel 1 and channel 2, respectively).

FREE1 - called when service on channel 1 is complete. Removes the message that was being processed and determines where it goes next. Also selects the next message for processing on channel 1 from the queue, if any are waiting.

FREE2 - similar in function to FREE1 except that it corresponds to channel 2.

ARIVAL - called when an IMS arrival occurs. Initiates message processing.

LEAVE - called when a message has completed all processing. Removes messages from the system.

REG1, REG2 - simulates the operation of the two message processing regions.

NXTMSG - called when a message processing region becomes available. Determines the next message to be processed in that region.

REUSE - called when an application program that is reuseable has completed the processing of a message. Searches the message queue for the next message that requires this program.

STATS - called at the end of simulation. Used to calculate various statistics from data accumulated within the system.

ENVIR1, ENVIR2 - used to generate external arrivals to channel 1 and channel 2, respectively.

Subroutines representing the third message processing region (REG3) and the third communication channel (CHNL3), along with any necessary program changes to accomodate these subroutines, will be used to model these additions to the existing system. REG3 and CHNL3 are similar in function to REG1 (REG2) and CHNL1 (CHNL2). A macroscopic flow diagram of the simulator is shown in Figure 2.2 and the program listing is in Appendix B.

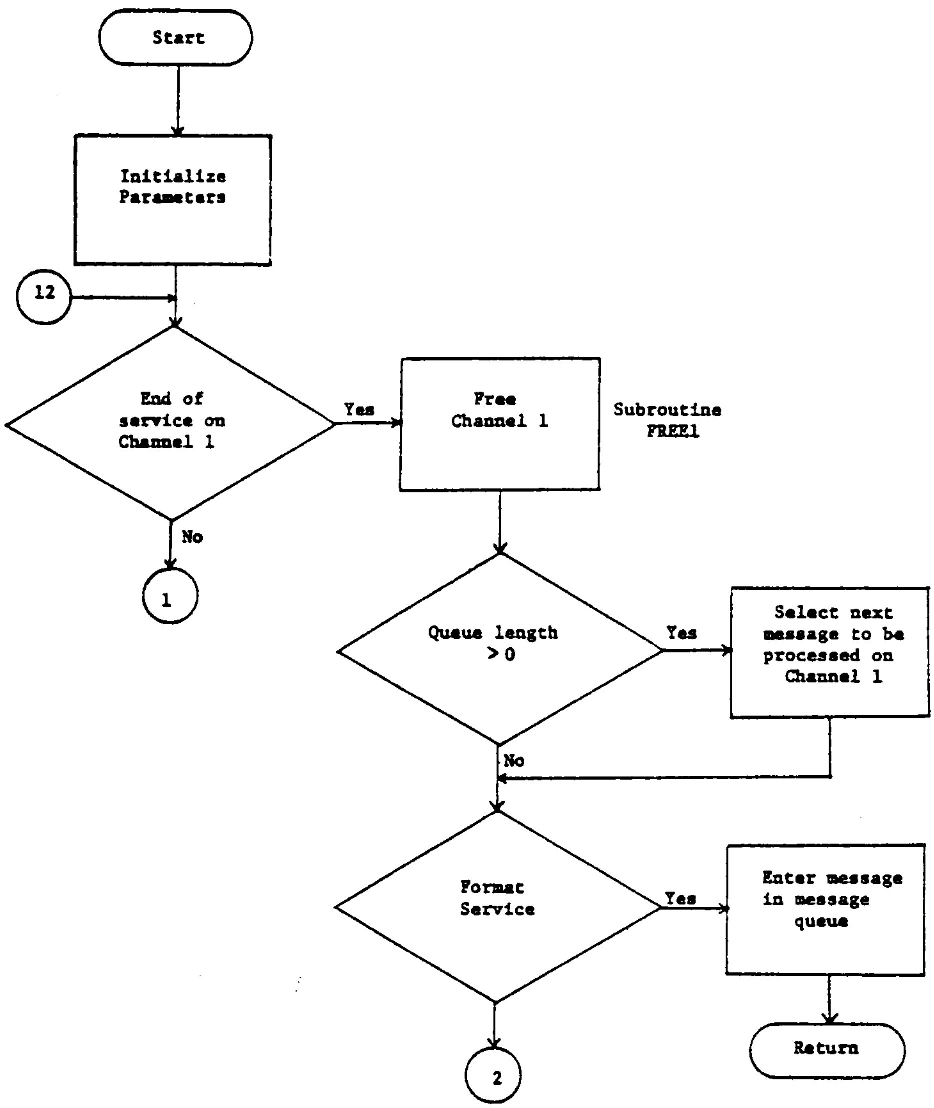


Figure 2.2 Macroscopic Flow Diagram of the Simulation Model

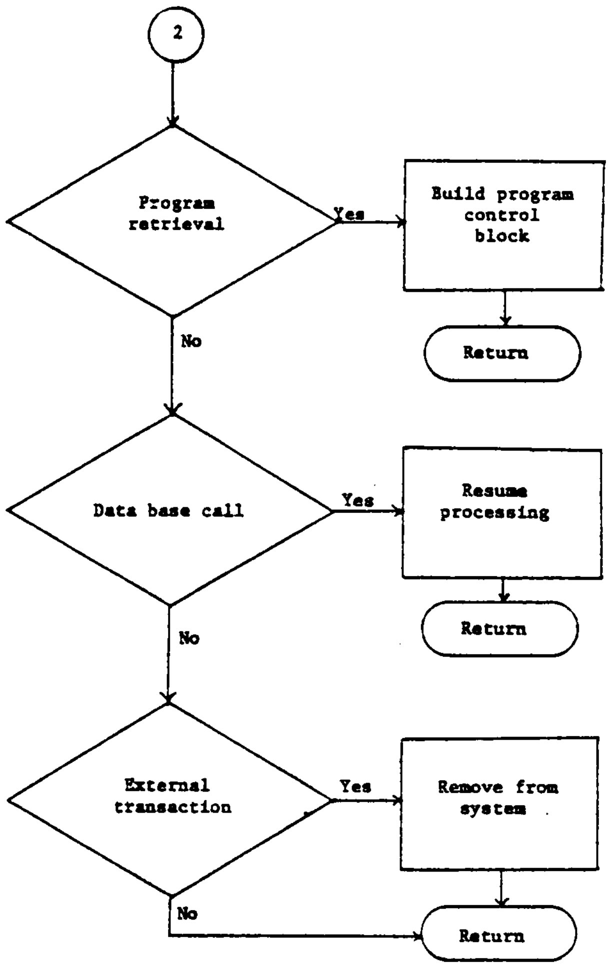


Figure 2.2 Macroscopic Flow Diagram of the Simulation Model (Continued)

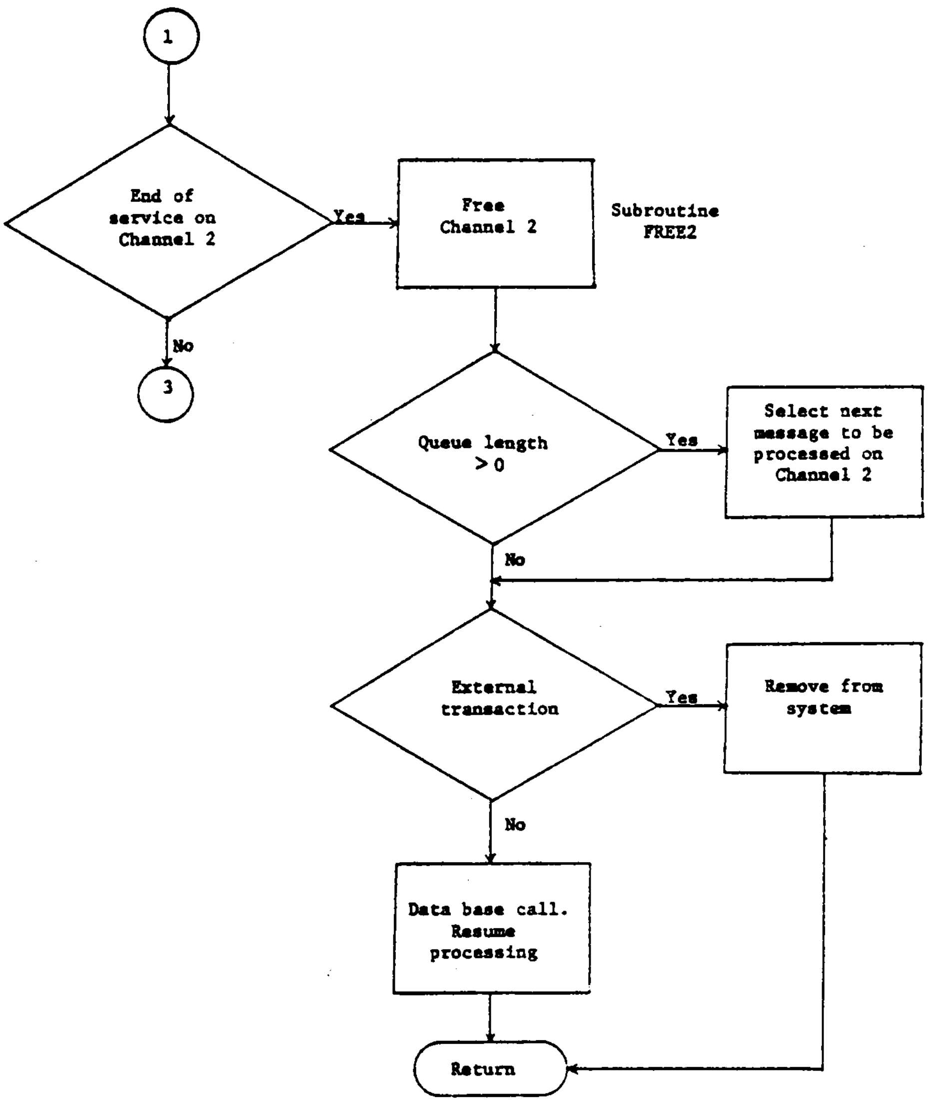


Figure 2.2 Macroscopic Flow Diagram of the Simulation Model
(Continued)

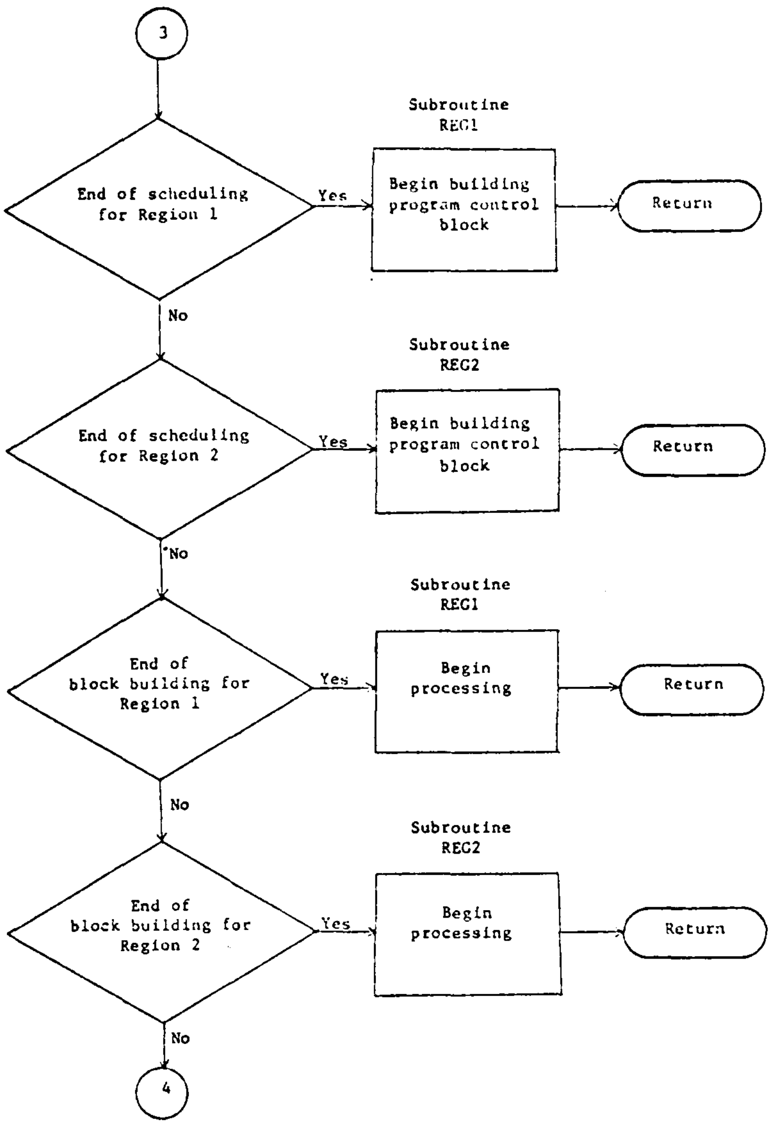


Figure 2.2 Macroscopic Flow Diagram of the Simulation Model (Continued)

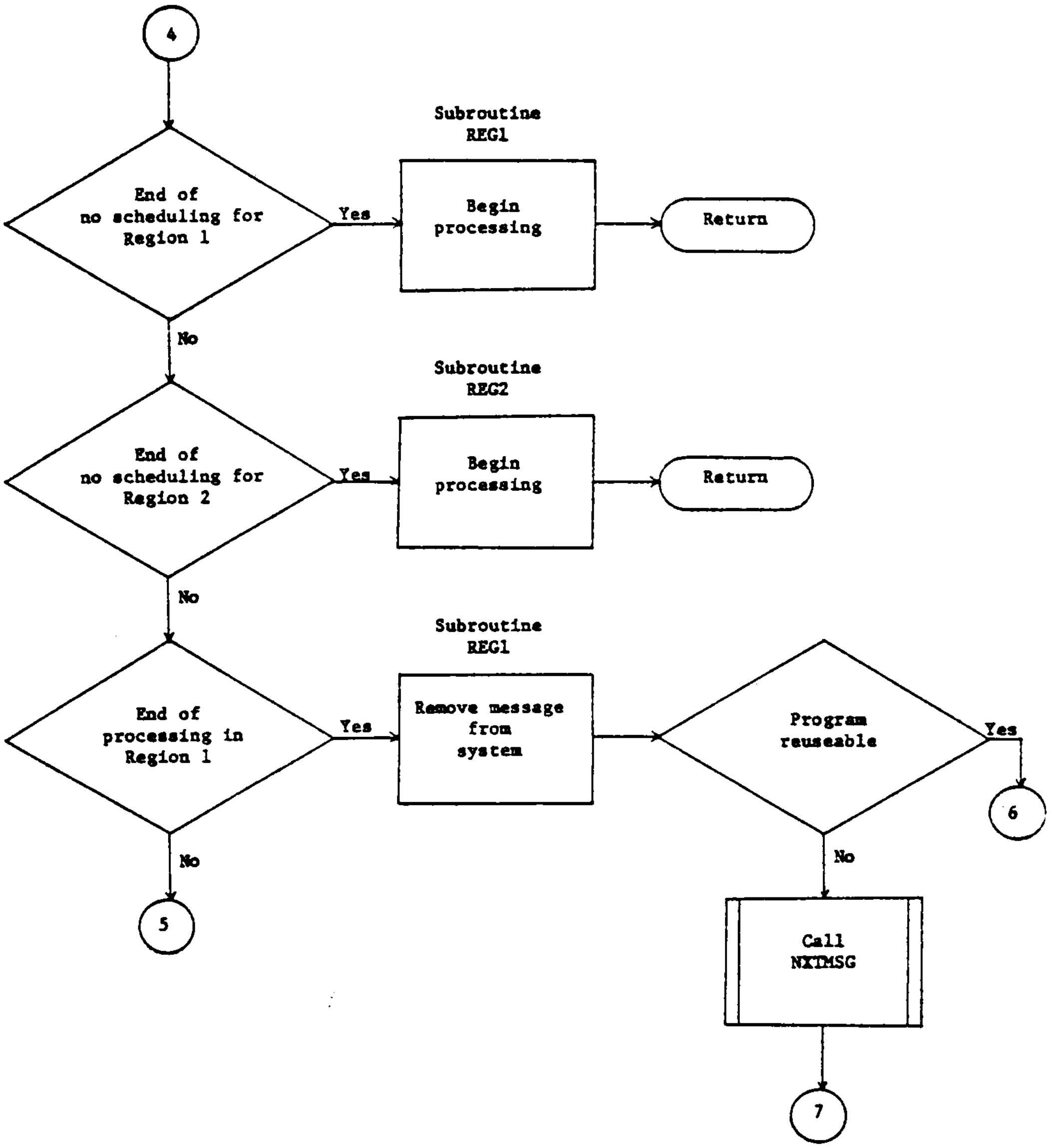


Figure 2.2 Macroscopic Flow Diagram of the Simulation Model (Continued)

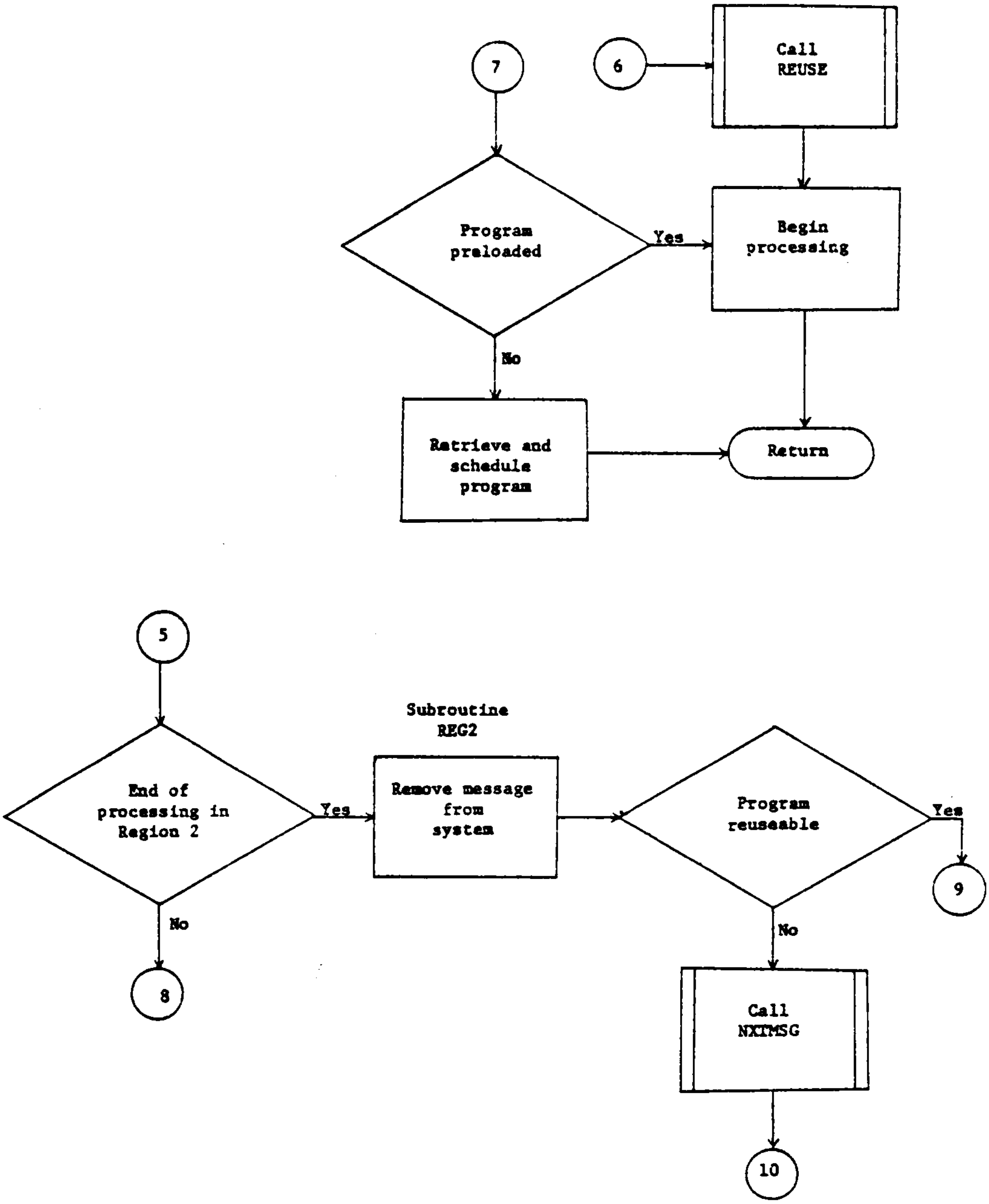


Figure 2.2 Macroscopic Flow Diagram of the Simulation Model (Continued)

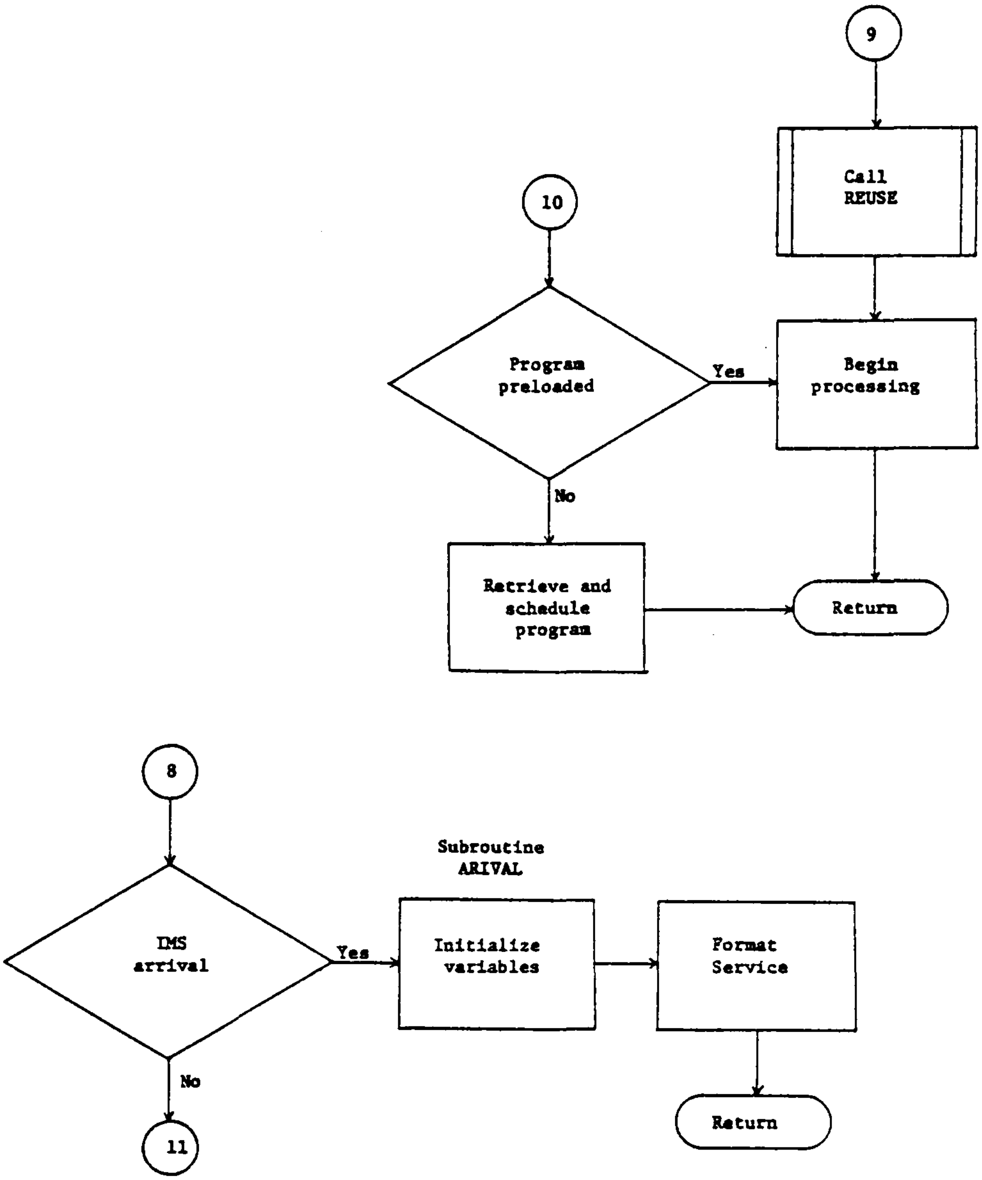


Figure 2.2 Macroscopic Flow Diagram of the Simulation Model (Continued)

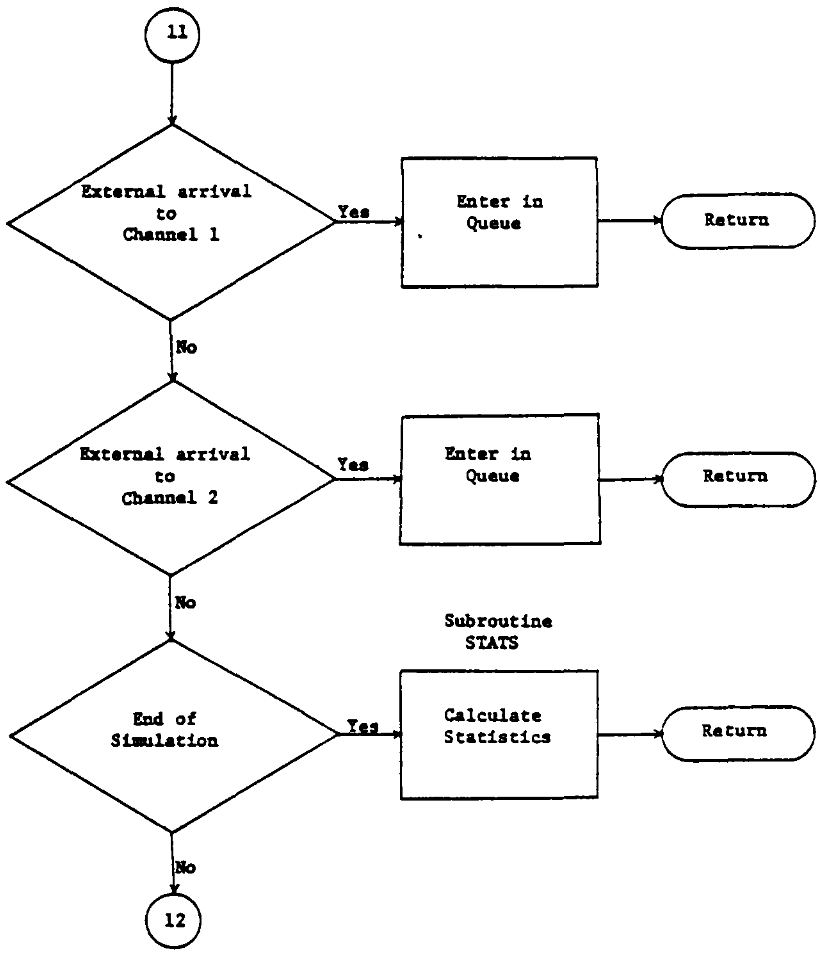


Figure 2.2 Macroscopic Flow Diagram of the Simulation Model (Concluded)

2.5 Input Data

A substantial amount of data collection and analysis was necessary to obtain data on message arrivals to the system and attributes of the message types under consideration. This section describes the data gathering procedure and the input data required.

Twenty-seven high volume message types representing more than 80% of the daily system load were selected for this analysis. Each of these message types has a number of attributes associated with it including:

1. Message type (1-27)
2. Message class
3. Normal priority in each region
4. Limit count
5. Limit priority
6. Application program required for processing
7. 1 if program is preloaded, 0 otherwise
8. 1 if program is reuseable, 0 otherwise
9. 1 if program must process serially, 0 otherwise
10. Region processing time
11. Data base processing time
12. Data bases required for calls
13. Processing limit.

The region processing time, the elapsed time for data base calls and the data bases that these calls go to were obtained from IMS Monitors and statistical reports. Values for the remaining attributes were obtained from the Systems Development Department at VPI&SU. These attributes are used throughout the simulation for various purposes, primarily for program control. All programs involved are reuseable and must process serially.

Since this is a next event simulation model, a process generator representing message interarrival times to the system is developed. Three months (66 working days) of data for hourly message volume for each day was available from IMS statistical reports. These reports yield hourly arrival summaries only; no information being available for interarrival times. In other words, the number of each type of message arriving in any one hour period was known, but direct data on the distribution of interarrival times is not available. The major difficulty was encountered in attempting to go from the distribution of arrivals per hour to the interarrival distribution. An attempt was made to fit known probability mass functions to the observed data with the hope of generating the corresponding continuous distribution of interarrival time. None of the distributions tried gave a fit that could be considered good or used with any degree of confidence.

Obviously, another method was needed to determine the distribution of arrivals to the system. A computer program was developed to synthesize arrivals from the available data. The cumulative distribution function of the number of arrivals per hour was generated from the observed data. A uniformly distributed random variate between 0 and 1 was generated to determine the number of arrivals of that message type for that particular hour from the cumulative distribution. These arrivals were then distributed uniformly within the hour. The algorithm employed can be described as follows:

1. Read the observed frequencies of message type J in hour K, OBS(J,K).

2. Calculate the number of times each observed frequency occurs, FREQ(I).

3. Accumulate the cumulative frequencies and calculate the cumulative probabilities,

   CUM = CUM + FREQ(I)

   PROB(I) = CUM/AN

   where AN = number of days in the sample.

4. Generate R, a uniformly distributed random variate on the interval [0,1].

5. Determine the frequency that this probability R corresponds to, Y = F(R).

6. Distribute these Y arrivals uniformly on the hour K.

All of these arrivals were sorted in ascending order and stored in a disk data file. This sequence of arrivals was used as arrival input data to the simulator for all runs. A listing of this program and a sample of the output is given in Appendix A.

CHAPTER III

VALIDATION OF THE SIMULATION MODEL

3.1 Introduction

Before the simulation model can be used for analytic purposes, it must be validated to the extent possible. The validation was carried out using data from IMS Monitors and statistics reports as compared against output from the simulation model. This chapter presents, in detail, the validation process that was performed.

3.2 The Validation Process

The validation process was performed in three phases:

1. Verfication of the structure of the simulation model
2. No load validation
3. Normal load validation.

Each of these phases will be discussed.

The verification phase was necessary to determine if the simulator represents the real system as it has been conceptualized in Chapter II. Messages were brought into the system at predefined intervals and their progress through the system closely monitored. In effect, the system was totally deterministic and the performance of the system and progress of any transaction within the system could be predicted. Any deviation from this predetermined flow would indicate an inconsistency between the simulator and the real system. When these bugs had been eliminated, the simulator accurately represented the transaction flow through the real system and the second phase of the validation could begin.

The first phase of the validation established the accuracy of modeling transaction flow through the system. The second phase was necessary to validate the timing of each message type under consideration in a no load situation. A no load situation exists when no external processing is occurring and all IMS processing is initiated from one terminal. This no load validation will establish a base from which the external loads on the communication channels can be determined through experimentation. It was therefore necessary to collect IMS data in a situation where there was no external interference. Ordinarily, the time from 8:00 a.m. until 4:00 p.m. on Sundays is reserved for equipment maintenance and testing, and normal processing of user programs is suspended. This would be an ideal time to activate IMS and collect data in this no load situation. With the cooperation and assistance of the Virginia Tech Computing Center, IMS was activated early one Sunday morning. Each of the transaction types under consideration was entered and allowed to fully process and exit the system before entering another. The monitor run for this test would reflect pure IMS processing with no external influence. This situation was simulated and the model tuned. The ratio of actual to simulated response time was used as a measure of difference. When this ratio is close to one, the simulation model accurately represents the real operation. The results of this phase of the validation is presented in Table 3.1.

The normal load validation was the most difficult phase of the analysis. In this phase it was necessary to estimate the external load on the communication channels through experimentation. The only data available on the external load was an estimate that external calls

Table 3.1 No Load Response Time Validation.

Message Type	Response Time (Sec)		Ratio A/S
	Actual	Simulated	
1	1.57	1.57	0.9987
2	0.28	0.28	1.0072
3	0.26	0.26	1.0000
4	0.35	0.35	1.0000
5	0.46	0.45	1.0132
6	0.23	0.22	1.0222
7	0.36	0.36	1.0084
8	0.20	0.20	1.0204
9	0.22	0.22	1.0138
10	0.21	0.21	0.9859
11	0.17	0.17	0.9272
12	0.30	0.30	1.0152
13	0.40	0.40	1.0010
14	0.30	0.29	1.0309
15	1.72	1.70	1.0130
16	0.20	0.20	0.9950
17	6.83	6.82	1.0019
18	0.24	0.24	1.0042
19	0.25	0.25	1.0121
20	0.52	0.52	0.9955
21	0.25	0.25	1.0013
22	0.20	0.20	0.9756
23	12.81	12.82	0.9993

Table 3.1 (Continued)

Message Type	Response Time (Sec)		Ratio A/S
	Actual	Simulated	
24	1.28	1.27	1.0079
25	0.50	0.50	0.9956
26	0.42	0.42	0.9929

outnumbered IMS calls by approximately 18 to 1. Using the ratio, it was assumed that the external arrivals could be generated exponentially with the mean of 18600 per hour on each channel. When an external arrival to one of the channels occurs, it is entered in the queue for that channel, processed for a specified (constant) service time, and then removed from the system. The service time for each channel was determined experimentally. Various combinations of service times for each channel were used in an effort to find the two values which best duplicated normal load conditions. The preliminary results of the experimentation with these service rates is presented in Table 3.2. The notation is as follows:

Q_1 = Service rate on channel 1.

Q_2 = Service rate on channel 2.

$$\bar{R} = \frac{\Sigma f_i R_i}{\Sigma f_i}$$

$$S = \sqrt{\frac{\Sigma f_i (R_i - 1)^2}{\Sigma f_i}}$$

where f_i = frequency of message type i and $R_i = \frac{\text{actual}}{\text{simulated}}$ for message type i.

From these preliminary results, the values for combinations of Q_1 and Q_2 ($Q_1 - Q_2$) that seemed most promising were: 125-130, 125-120, 110-130, 110-120, 100-130, 100-120, 90-130, and 90-120. Four 5 minute runs were made using each of these pairs of values and a different starting time and random number seed. These four replications were used to calculate the overall $\bar{R}$ and S. These results are presented in Table 3.3.

Table 3.2 Preliminary Normal Load Validation Results.

Q_1	Q_2	$\overline{R}$	S
90	120	1.50	2.45
90	130	1.12	1.56
100	110	1.72	5.27
100	120	1.23	1.40
100	130	0.99	1.25
100	140	0.56	0.84
110	120	1.29	1.92
110	130	1.36	2.22
125	100	1.46	4.68
125	120	1.33	2.15
125	130	1.04	1.75
125	140	0.65	1.76

Table 3.3 Replication of Five Minute Validation Runs.

Q_1	Q_2	S	$\bar{R}$	F	Overall
125	130	1.75 1.89 0.37 0.42	1.04 1.17 0.62 0.78	98 67 55 68 288	$\bar{R}$ = .93 S = .22
125	120	2.15 1.09 0.33 0.49	1.33 0.94 0.67 0.88	98 67 55 68 288	$\bar{R}$ = 1.01 S = .25
110	130	2.22 1.62 0.25 0.52	1.36 1.19 0.80 0.58	98 67 55 69 289	$\bar{R}$ = 1.03 S = .32
110	120	1.92 1.86 0.27 1.80	1.29 1.31 0.87 1.02	98 67 55 69 289	$\bar{R}$ = 1.15 S = .23
100	130	1.25 1.57 0.29 0.54	.99 1.17 .87 0.58	98 67 55 67 287	$\bar{R}$ = .92 S = .22
100	120	1.40 1.84 0.33 1.84	1.23 1.22 0.94 1.23	98 67 55 67 287	$\bar{R}$ = 1.17 S = .21
90	130	1.56 1.36 0.24 11.64	1.12 0.99 .90 2.82	98 67 55 66 286	$\bar{R}$ = 1.44 S = .88
90	120	2.45 1.88	1.50 1.35	98 67	$\bar{R}$ = 1.19 S = .35

Table 3.3 (Continued)

Q_1	Q_2	S	$\overline{R}$	F	Overall
		0.33 0.66	0.98 0.79	55 69 289	
		GRAND TOTAL		2303	$\overline{R}$ = 1.10 S = .19

The next step was to run the simulator for thirty minutes to further verify the values. The results, which are given in Table 3.4, were not at all expected. This indicated that messages were taking too long in the system due to system loading that did not occur during the five minute runs. The normal load validation would have to be carried out again using 30 minute simulation runs.

The best results from these thirty minute simulation runs were obtained with 50-40. The next step was to run the simulator for different time periods during the day to determine the overall $\bar{R}$ and S. These results are given in Table 3.5. The thirty minute periods with $\bar{R}$ close to one will be used or the analysis runs. These statistics, $\bar{R}$ and S, were also calculated for data obtained from the actual system. This data is presented in Table 3.6 for comparison purposes. Four days were used as a sample and the following notation is used:

R_{ij} = Resonse time (actual) for message type j on day i.

f_{ij} = Frequency of message type j on day i.

$$A_j = \frac{\sum_i f_{ij} R_{ij}}{\sum_i f_{ij}} \quad \forall i,j$$

$$\bar{R}_i = \frac{\sum_i f_{ij} \frac{A_j}{R_{ij}}}{\sum_i f_{ij}} \quad \forall i,j$$

Table 3.4 Results from 30 Minute Simulation Runs.

Q_1	Q_2	$\overline{R}$	S
70	80	0.59	0.71
70	90	0.50	0.63
70	70	0.69	0.86
40	40	0.91	0.98
40	50	0.88	1.12
50	50	0.84	0.99
50	40	0.93	0.99
90	110	0.67	1.01
100	120	0.49	0.90
110	130	0.27	0.78

Table 3.5 30 Minute Simulation Runs With 50-40.

Start	$\overline{R}$	S	f
10:30	0.98	0.97	459
11:00	1.18	0.80	345
11:30	1.59	1.53	319
12:00	2.67	1.91	115
13:00	1.76	1.18	338
14:00	1.06	1.11	363
14:30	2.90	2.39	341
15:00	2.07	2.53	346
15:30	1.04	1.31	332
Overall	1.74	.88	2958

Table 3.6 Calculation of $\bar{R}$ and S for Actual Data

j	f_{1j}	R_{1j} (sec)	f_{2j}	R_{2j} (sec)	f_{3j}	R_{3j} (sec)	f_{4j}	R_{4j} (sec)	A_j	A_j/R_{1j}	A_j/R_{2j}	A_j/R_{3j}	A_j/R_{4j}
1	26	0.82	80	1.55	36	1.13	122	0.96	1.15	1.40	0.74	1.02	1.20
2	138	2.36	16	2.49	79	6.29	31	1.96	3.50	1.48	1.41	0.56	1.79
3	44	1.23	9	1.42	30	2.56	25	1.23	1.62	1.32	1.14	0.63	1.32
4	1	3.20	1	0.40	4	1.03	3	0.53	1.03	0.32	2.58	1.00	1.94
5	378	1.96	112	1.57	102	3.45	46	3.09	2.21	1.13	1.41	0.64	0.72
6	77	1.20	19	1.91	15	0.82	33	0.64	1.13	0.94	0.59	1.38	1.77
7	86	0.75	81	1.28	145	1.16	406	0.65	0.84	1.12	0.66	0.72	1.29
9*	121	1.00	23	0.83	20	0.42	51	1.34	1.01	1.01	1.22	2.40	0.75
10	161	0.97	345	1.40	33	1.01	81	0.66	1.17	1.21	0.84	1.16	1.77
11	173	0.78	481	1.29	139	1.54	212	0.53	1.08	1.38	0.84	0.70	2.04
12	180	1.06	532	1.32	207	1.03	355	0.61	1.04	0.98	1.79	1.01	1.70
13	94	0.82	86	1.25	87	1.17	115	0.83	1.00	1.22	0.80	0.85	1.20
14	248	0.53	194	0.81	201	0.94	260	1.04	0.83	1.57	1.02	0.88	0.80
15	134	2.11	248	2.26	184	2.07	277	2.05	2.13	1.01	0.94	1.03	1.04
16	470	0.59	214	0.99	243	1.68	102	1.10	0.98	1.66	0.99	0.58	0.89
17	137	0.92	50	1.11	85	1.40	110	2.06	1.38	1.50	1.24	0.99	0.67
18	3	0.58	2	0.38	2	0.26	4	0.79	0.56	0.97	1.47	2.15	0.71
19	4	0.76	2	0.49	3	1.78	7	1.16	1.09	1.43	2.22	0.61	0.94
20	16	1.06	228	1.29	131	1.25	164	1.29	1.27	1.19	0.98	1.02	0.98
21	22	1.72	228	1.34	153	1.75	149	1.53	1.52	0.88	1.13	0.87	0.99
22	25	2.75	6	2.24	9	1.43	2	0.63	2.29	0.83	1.02	1.60	3.63
23	94	0.95	182	2.74	144	1.65	113	1.87	1.95	2.05	0.71	1.18	1.04
24	166	1.71	208	1.91	244	1.85	295	2.14	1.93	1.13	1.01	1.04	0.90
25	382	1.41	623	1.76	687	1.77	642	1.13	1.53	1.09	0.87	0.86	1.35
26	42	1.16	15	1.08	24	0.77	31	1.00	1.02	0.88	0.94	1.32	1.32
	3222		3985		3007		3636			$\bar{R}_1 = 1.29$	$\bar{R}_2 = 0.91$	$\bar{R}_3 = 0.89$	$\bar{R}_4 = 1.23$
$\bar{R} = 1.08$ S = 0.20													

* No data was available for message type 8.

$$\overline{R} = \frac{\sum_{i} \left(\sum_{j} f_{ij} \right) \overline{R}_i}{\sum_{i} \sum_{j} f_{ij}} \quad \forall i,j$$

$$S = \sqrt{\frac{\sum_{i} \left(\sum_{j} f_{ij} \right) (\overline{R}_i - 1)^2}{\sum_{i} \sum_{j} f_{ij}}} \quad \forall i,j$$

With the validation process complete, the simulation model can be considered as a valid representation of the real system. Various system alterations were made and the results obtained are discussed in the next chapter.

CHAPTER IV

ANALYSIS OF RESULTS

4.1 Introduction

This chapter presents the results obtained from the simulation model under various operating conditions. These test situations are:

1. Increasing IMS load.
2. Increasing external load.
3. Increasing both IMS and external loads.
4. Changing the message class ordering for the processing regions.
5. Varying the distribution of IMS data bases on the communication channels.
6. Adding a third processing region.
7. Adding a third communications channel.
8. Adding a third communications channel and a third processing region.

Each of these will be discussed separately. External load on the processing region is not considered since IMS has priority over other processing.

4.2 Existing Configuration

The IMS load on the existing configuration was increased to simulate peak load conditions (approximately 23000 messages per day) so that a base for the analysis could be established. The IMS load was then increased 10%, 25% and 50% to determine the effect on average overall response time. Similarly, the external load was increased 10%, 25% and 50%. Simultaneous increases in IMS and external loads of 10%, 25% and 50% were used to study the effect of increasing both loads. These results are given in Table 4.1 and shown graphically in Figure 4.1. As was expected, the overall average response time increased as the loads increased. The performance of the various configurations presented in subsequent sections will be compared against these values to determine if any improvement in system performance occurs.

4.3 Addition of a Third Processing Region

A third message processing region was added to the existing configuration and the following variations in system load analysed:

1. Increasing IMS load 10%, 25% and 50%.
2. Increasing external load 10%, 25% and 50%.
3. Simultaneously increasing IMS and external loads 10%, 25% and 50%.

The results for this test configuration are listed in Table 4.2 and illustrated in Figure 4.2.

The addition of the third processing region caused the overall average response to improve slightly over the existing configuration. However, this improvement is not significant.

4.4 Addition of a Third Communications Channel

A third data communications channel was added to the existing configuration and the effect on response time analyzed under the different load conditions. Again, the IMS and external loads were increased 10%, 25% and 50%, individually and simultaneously. Table

Table 4.1 Effect of Increased Loads on IMS Response Time for Existing Configuration.

% Increase	Response (Sec.)		
	IMS	External	Both
0 (Peak)	4.25	4.25	4.25
10	5.44	4.70	5.26
25	7.19	8.10	7.52
50	10.42	10.60	10.83

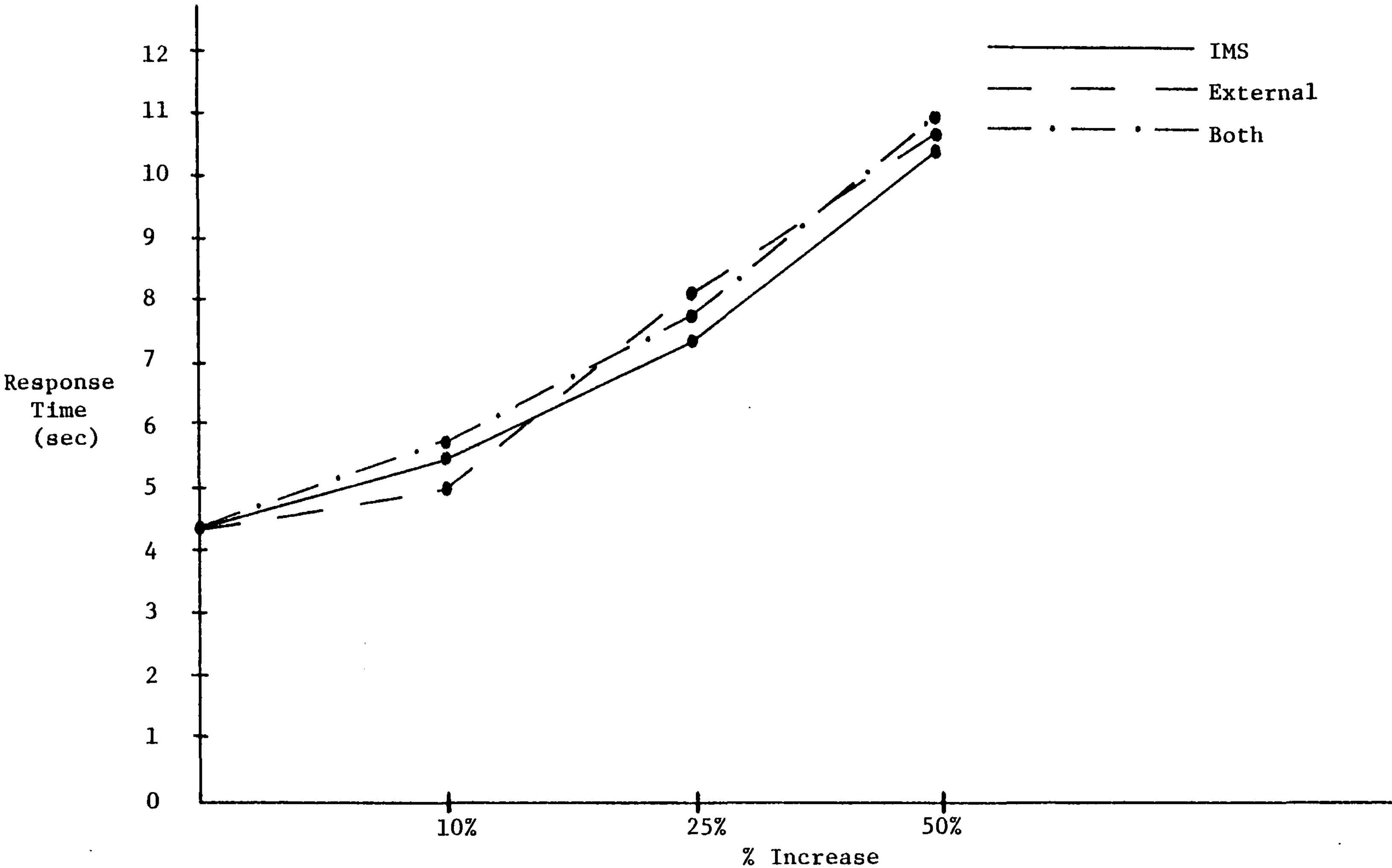


Figure 4.1 Effect of Increased Loads on IMS Response Time for Existing Configuration.

Table 4.2 Effect of Increased Loads on IMS Response Time for Configuration with Third Region.

% Increase	Response (Sec.)		
	IMS	External	Both
0 (Peak)	3.75	3.75	3.75
10	4.70	7.30	7.30
25	6.90	7.90	9.80
50	8.30	9.40	10.10

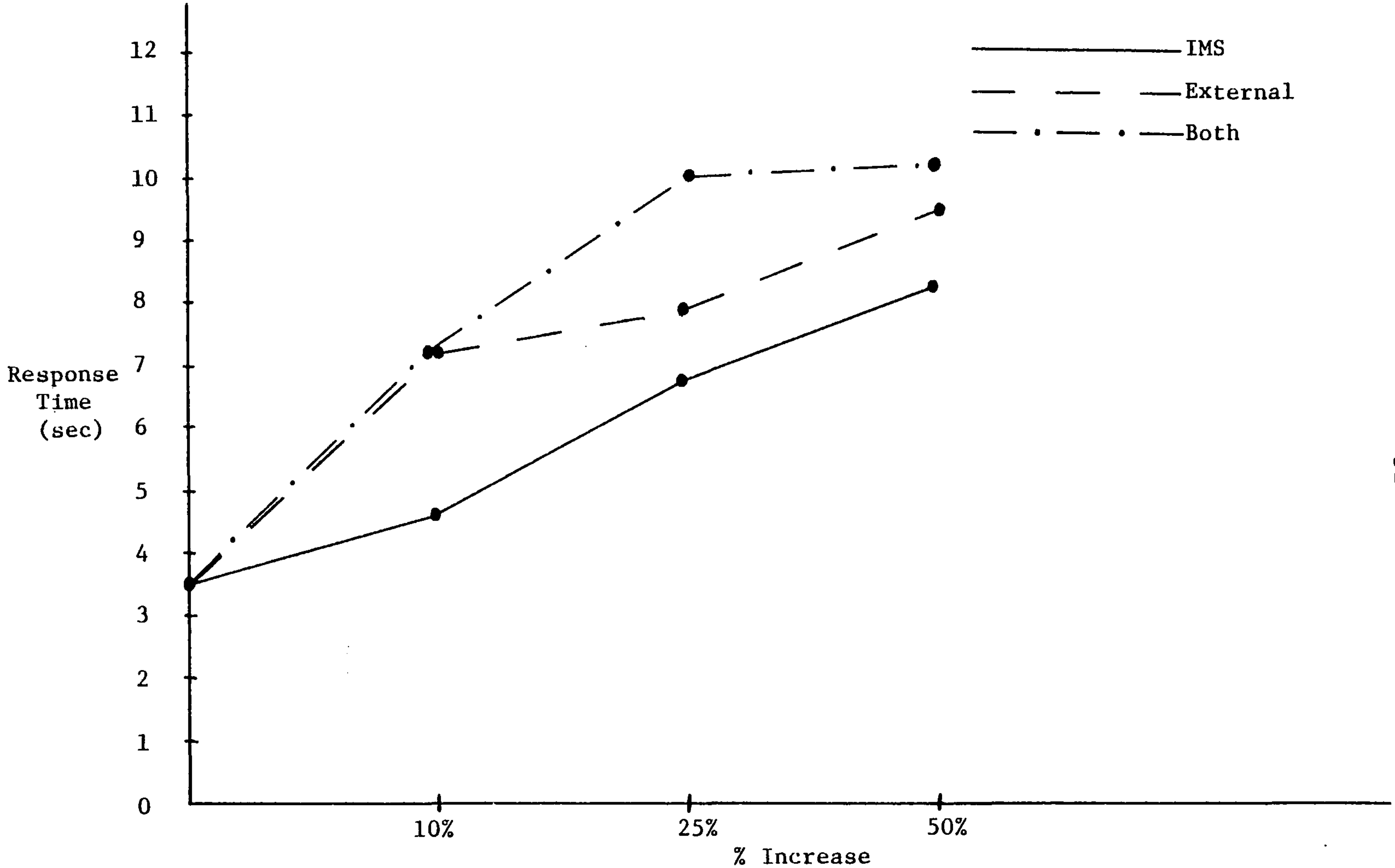


Figure 4.2 Effect of Increased Loads on IMS Response Time for Configuration with Third Region.

4.3 shows the data base configuration used for three channels as compared to the existing (two-channel) system. Also, two versions of this system configuration were considered: 1) a third, dedicated channel, and 2) a shared channel. The dedicated channel has no external load on it. For the shared channel, one third of the external loads on the other two channels was diverted to the third. Essentially, the overall external load was the same but distributed differently. The distribution was two-thirds of the total external load to each of the three channels.

The response times for the dedicated channel configuration under various load conditions are given in Table 4.4 and graphed in Figure 4.3. The response times for the case with the shared channel are given in Table 4.5 and illustrated in Figure 4.4. When the external load was redistributed to include the third channel, the response actually improved somewhat. This is attributed to the load on the original two channels being reduced, even though the overall external load is effectively the same.

4.5 Addition of Third Region and Third Channel

Since the independent addition of a third message processing region and a third communications channel improved system performance, it appeared reasonable to try adding both. The same percent increases in IMS and external load that were used to test the other configurations were also used to test this one. The results are presented in Table 4.6 and illustrated in Figure 4.5

Response time not only did not improve over the three channel

Table 4.3 Data Base Configurations

Data Base	2 Channels	3 Channels	Data Base	2 Channels	3 Channels
1	1	1	24	1	2
2	1	1	25	1	2
3	1	1	26	1	2
4	1	1	27	1	2
5	1	1	28	1	2
6	1	1	29	1	2
7	1	1	30	1	2
8	1	1	31	2	2
9	1	1	32	2	3
10	1	1	33	2	3
11	1	1	34	2	3
12	1	1	35	2	3
13	1	1	36	2	3
14	1	1	37	2	3
15	1	1	38	2	3
16	1	1	39	2	3
17	1	1	40	2	3
18	1	1	41	2	3
19	1	1			
20	1	1			
21	1	2			
22	1	2			
23	1	2			

Table 4.4 Effect of Increased Loads on IMS Response Time for Configuration with Third (Dedicated) Channel.

% Increase	Response (Sec.)		
	IMS	External	Both
0 (Peak)	1.80	1.80	1.80
10	1.81	1.82	1.86
25	1.98	1.88	2.10
50	2.20	2.84	6.08

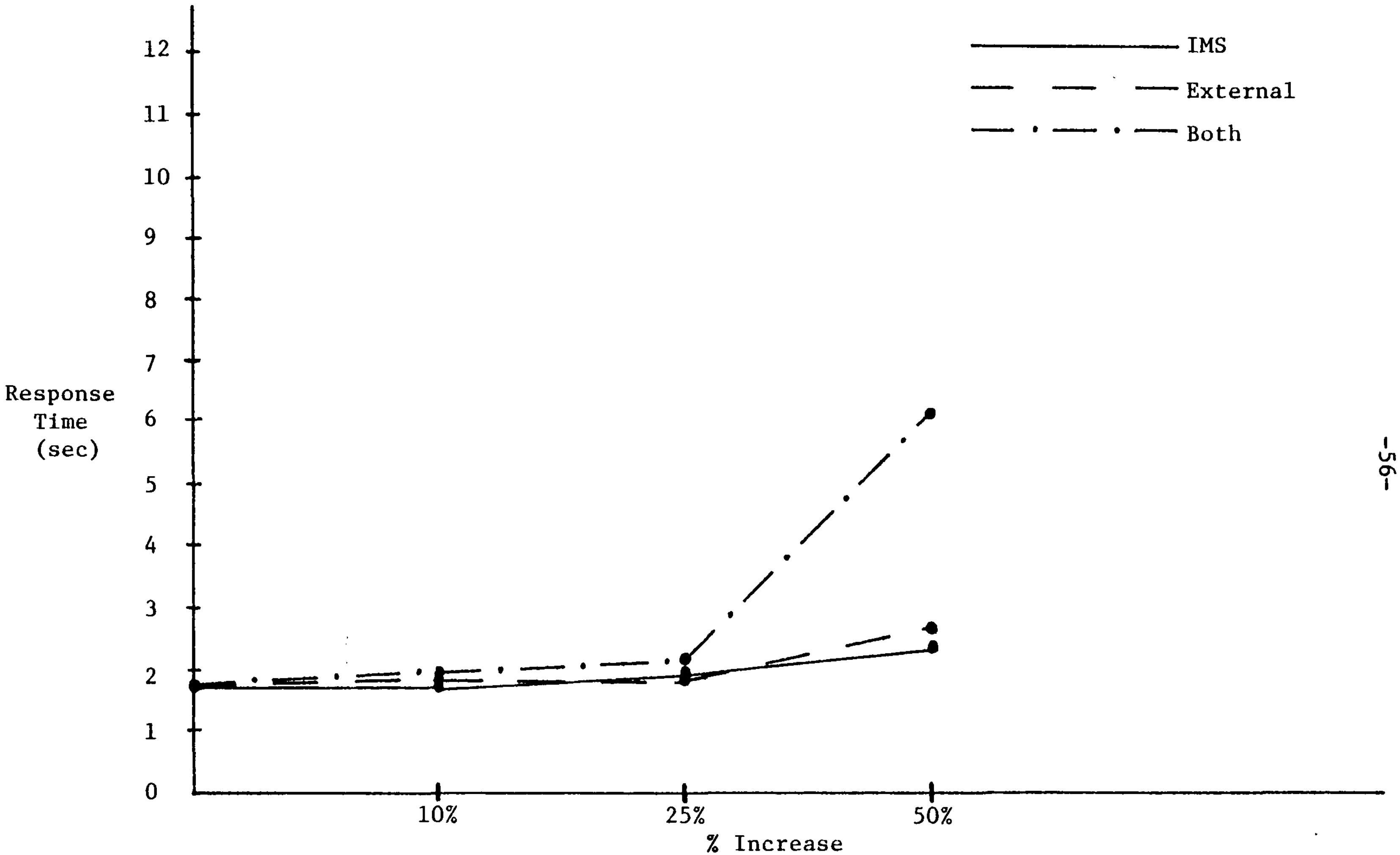


Figure 4.3 Effect of Increased Loads on IMS Response Time for Configuration with Third (Dedicated) Channel.

Table 4.5 Effect of Increased Loads on IMS Response Time for Configuration with Third (Shared) Channel.

% Increase	Response (Sec.)		
	IMS	External	Both
0 (Peak)	1.68	1.68	1.68
10	1.70	1.71	1.73
25	1.82	1.75	1.88
50	5.64	1.79	7.75

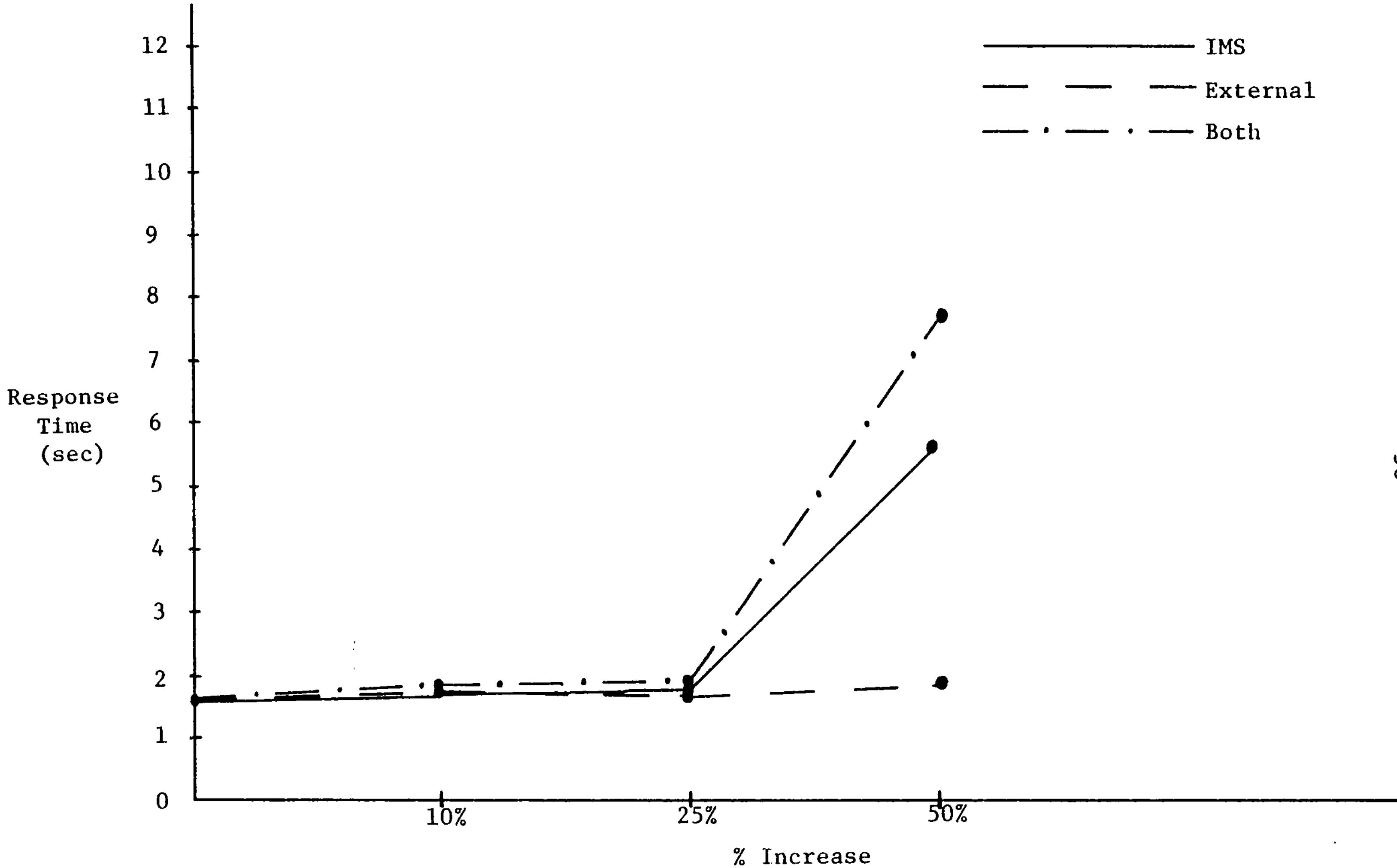


Figure 4.4 Effect of Increased Loads on IMS Response Time for Configuration with Third (Shared) Channel.

Table 4.6 Effect of Increased Loads on IMS Response Time for Configuration with Third Region and Third Channel

% Increase	Response (Sec.)		
	IMS	External	Both
0 (Peak)	1.93	1.93	1.93
10	2.01	1.96	2.07
25	2.22	1.99	2.47
50	5.68	2.11	6.45

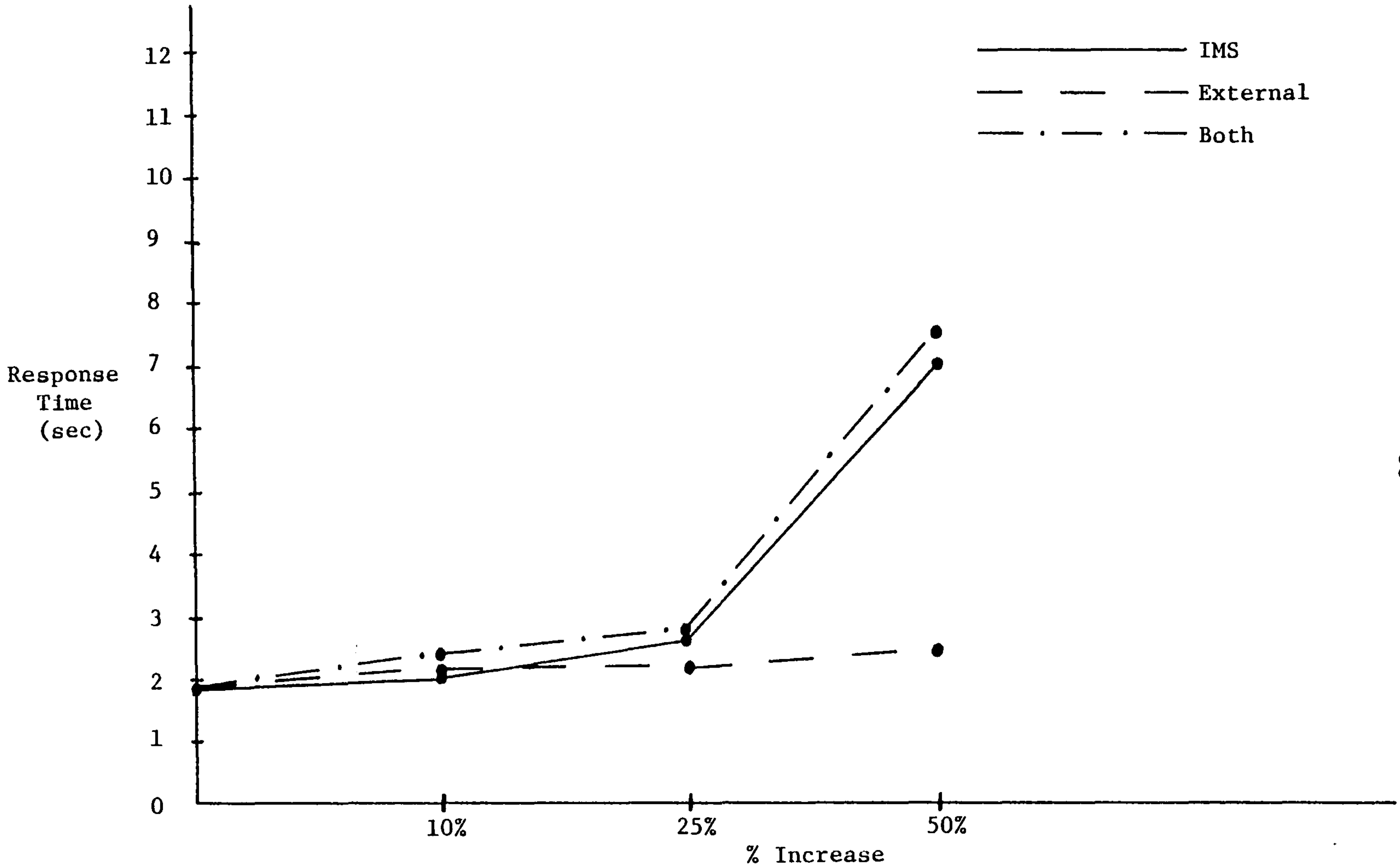


Figure 4.5 Effect of Increased Loads on IMS Response Time for Configuration with Third Region and Third Channel.

configuration, but actually deteriorated somewhat. This was due to the communications channel representing a feedback loop. This phenomenon can be illustrated by the following example. Consider the system shown in Figure 4.6a and two transactions as given in Table 4.7 (P represents region processing time, C represents channel time). The transactions enter the system and process in the sequence in Table 4.7. Processing in the region cannot resume until processing on the channel has been completed. For the system in Figure 4.6a, the time in the system for transaction 1 is 16 and for transaction 2 is 22 + 16 = 38. The average response time for the two transactions is 27.

Now consider the system in Figure 4.6b. Using the same sequence from Table 4.7, the following results are obtained (Table 4.8). The time in the system for both transaction 1 and transaction 2 is 31 given an average response time of 31. This is an increase in average response over the system in Figure 4.6a, even with the addition of the second region. If, in the simulation model, the programs processing in two or more of the regions require data on the same channel, then its delay can be expected and average overall response time would increase. This situation could possibly be alleviated by experimenting with different distributions of data bases to channels to eliminate or lessen the chance of the bottleneck occurring.

4.6 Changing Message Class Ordering

The message class ordering for the two regions of the existing configuration were varied to determing if this had any effect on response. The results are given in Table 4.9. From the simulation

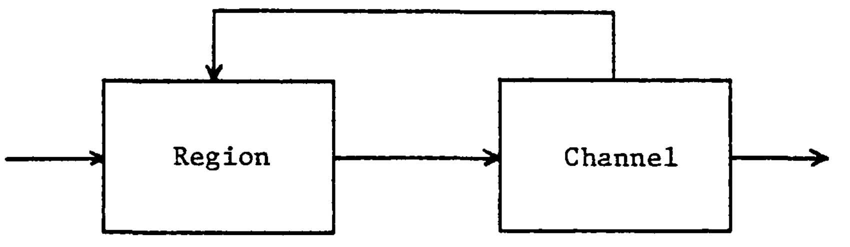


Figure 4.6a Feedback Loop with One Region and One Channel

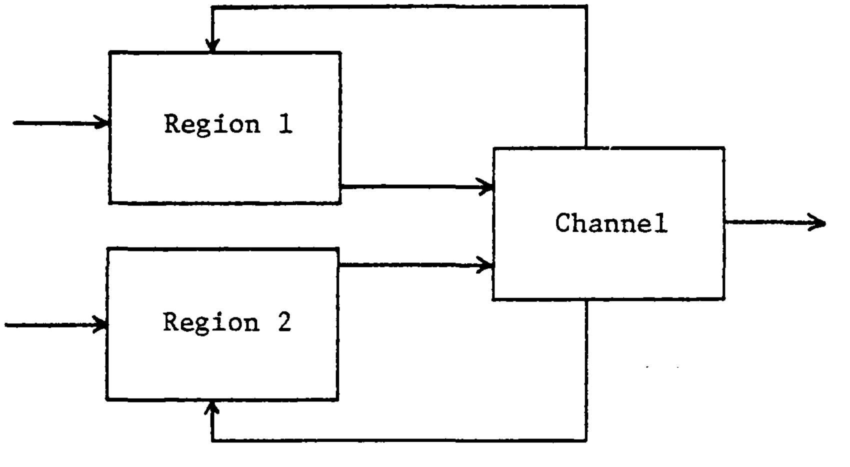


Figure 4.6b Feedback Loop with Two Regions and One Channel

Table 4.7 Example Transactions

Transaction 1	Transaction 2
P = 2	P = 1
C = 3	C = 15
P = 2	P = 1
C = 8	C = 3
P = 1	P = 1
16	C = 1
	22

P = Processing in Region

C = Processing on Channel

Table 4.8 Example Results

Time	1	2	3	...	16	17	18	19	20	21	22	23	24	...	30	31	32
Transaction 1	P	P	Q	→	Q	C	C	C	P	P	Q	C	C	→	C	P	→
Transaction 2	P	C	C	→	C	P	Q	Q	C	C	C	P	Q	→	Q	C	→

P = Processing in Region

C = Processing on Channel

Q = Waiting in Queue

Table 4.9 Effect of Changing Message Class Ordering on IMS Response Time

Configuration (Region 1 - Region 2)	Response (Sec.)
3,2,1 - 2,1,3 (Existing)	4.25
3,2,1 - 1,2,3	6.41
3,2,1 - 3,2,1	6.07
3,2,1 - 2,3,1	5.90
3,2,1 - 1,3,2	5.90
3,2,1 - 3,1,2	6.08

results, it is evident that no advantage can be gained from changing the message class orderings.

4.7 Changing Distribution of Data Bases

The existing data base distribution was changed to determine if this would have any effect on system performance. Two data base configurations were used. One configuration simply switched the data bases on channel 1 with those on channel 2. The other configuration had the higher volume data bases on channel 2 and those with less use on channel 1. The format service remained on channel 1. Since external processing does not access IMS data bases, the external loads on the channels were not altered. The data base configurations are given in Table 4.10 and the results are given in Table 4.11. No improvement was realized from either of the configurations. This does not mean, however, that a more efficient configuration does not exist.

4.8 Regression Models

The output obtained from the simulation models was used to develop regression models representing the different configurations. These regression models were developed using the General Linear Models (GLM) procedure of the Statistical Analysis System (SAS). Accurate regression models can be used to predict system performance as a function of any combination of IMS and external loads, but caution is needed to insure that the models are not used outside the range of the data. A number of possible regression models were investigated. The models

Table 4.10 Data Base Configurations

Data Base	Channel		
	Existing	Configuration 1	Configuration 2
1	1	2	1
2	1	2	1
3	1	2	1
4	1	2	1
5	1	2	1
6	1	2	1
7	1	2	2
8	1	2	1
9	1	2	1
10	1	2	1
11	1	2	1
12	1	2	1
13	1	2	1
14	1	2	1
15	1	2	1
16	1	2	1
17	1	2	1
18	1	2	2
19	1	2	1
20	1	2	1
21	1	2	1
22	1	2	2
23	1	2	1

Table 4.10 (Continued)

Data Base	Channel		
	Existing	Configuration 1	Configuration 2
24	1	2	1
25	1	2	1
26	1	2	1
27	1	2	1
28	1	2	1
29	1	2	1
30	1	2	2
31	2	1	2
32	2	1	1
33	2	1	2
34	2	1	2
35	2	1	1
36	2	1	1
37	2	1	1
38	2	1	2
39	2	1	2
40	2	1	2
41	2	1	1

Table 4.11 Effect of Changing Data Base Configuration on Response Time.

Configuration	Response (Sec.)
Existing	4.25
Configuration 1	6.40
Configuration 2	5.40

Table 4.12 Regression Models

Configuration	Model	R^2	*Average % Error
Existing	$Y = -21.559 + 15.116X_2^2 + 35.006X_1 - 24.309X_1X_2$	.851	8.53
3 Regions	$Y = -67.842 - 9.581X_1^2 - 21.462X_2^2 + 35.872X_1 + 69.206X_2 - 1.985X_1^2X_2^2$	.859	7.92
3 Channels	$Y = 53.990 + 24.067X_1^2 + 10.223X_2^2 - 58.188X_1 - 29.501X_2 + 1.578X_1^2X_2^2$	.965	13.94
3 Channels/ 3 Regions	$Y = 33.216 + 21.313X_1^2 + 2.327X_2^2 - 48.126X_1 - 7.174X_2 + .578X_1^2X_2^2$	.979	6.21

$$\text{* Average \% Error} = \frac{1}{n}\sum\frac{|Y_i - \hat{Y}_i|}{Y_i} \times 100$$

where Y_i = Actual Value

$\hat{Y}_i$ = Model Value

yielding the best results are given in Table 4.12. The independent variables are X_1 and X_2, where X_1 represents IMS load and X_2 represents external load. These are expressed as a ratio of proposed load to current load. For example, a 15% increase in IMS load implies $X_1 = 1.15$, and so on. The dependent variable, Y, represents IMS response time in seconds. The data used to generate these models as well as the SAS output are given in Appendix C.

4.9 Summary of Results

The results obtained from the simulation models indicate that the addition of the third communications channel resulted in a significant improvement in IMS response. Unless the costs associated with implementing a third communications channel are indisputably prohibitive, the addition of the third channel would be the best improvement to the system. Based on the simulation results, the addition of the third channel would yield the greatest long-run benefits. The other additions give an improvement in response time, but the addition of the third channel results in the longest sustained improvement. A comprehensive summary of results for all the system configurations discussed is given in Table 4.13.

Table 4.13 Summary of Results

		Existing	Third Region	Third Channel (Dedicated)	Third Channel (Shared)	Three Regions Three Channels
	Peak	4.25	3.57	1.80	1.68	1.93
IMS	10%	5.44	4.70	1.81	1.70	2.01
	25%	7.19	6.90	1.98	1.82	2.22
	50%	10.42	8.30	2.20	5.64	5.68
External	10%	4.70	7.30	1.82	1.71	1.96
	25%	8.10	7.90	1.88	1.75	1.99
	50%	10.60	9.40	2.84	1.79	2.11
Both	10%	5.26	7.30	1.86	1.73	2.07
	25%	7.52	9.80	2.10	1.88	2.47
	50%	10.83	10.10	6.08	7.75	6.45

CHAPTER V

CONCLUSIONS AND EXTENSIONS

5.1 Introduction

The development of the simulation model was presented in Chapter II and the validation procedure was discussed in Chapter III. Chapter IV contains an analysis of results and the development of associated regression models. This chapter presents the conclusions and suggests possible areas for further investigation.

5.2 Conclusions

The purpose of this thesis was to develop and validate a simulation model for IMS and use this model to analyze system performance under a variety of operating conditions. In particular, this thesis illustrated the utility derived from simulating IMS operation with a view toward achieving an improvement in system performance through:

(1) Adding a third processing region.

(2) Adding a third communications channel.

(3) Adding both a third region and a third channel.

(4) Redistributing data bases on the channels.

(5) Changing the message class configurations.

The research that has been presented supports the fact that simulation can be an effective method for analysis when data base management systems are involved. Because of the complexities of such systems, simulation can be a very utilitarian method of modeling, analysis and optimization. For example, the results obtained from the simulation models applied in this research indicate that the addition of a third communications channel to the system studied results in a significant improvement in response time. Even with a 100% increase in external load, the response time is comparable to a 25% increase in external load under the present system. Unless the costs associated with implementing a third communications channel were indisputably prohibitive, the addition of a third channel would be the best improvement, of those examined, to the system. The other system modifications which were investigated each resulted in an improvement in response time, but the addition of the third channel results in the greatest sustained improvement.

5.3 Extensions and Areas for Further Investigation

Several extensions of this research are possible. One would be to take a more microscopic view of the system. This could include an analysis of the scheduling algorithm employed to determine its effect on system performance, and an in-depth investigation of the priority scheme used in scheduling transactions.

Instead of aggregating IMS arrivals to the system, the arrivals generated by each terminal in the network could be investigated. If interarrival times can be obtained, the distribution of arrivals generated by each terminal can be determined.

Several additional areas for further investigation have been suggested as a consequence of this research. A more precise determination of the distribution of external arrivals is needed. For this research, external arrivals were assumed to be exponential with mean equal to 18 times the IMS rate. The estimates used herein gave favorable results, but this does not necessarily mean that these estimates are correct. Hardware and/or software monitors could possibly be used to obtain a better estimate of external arrival rates.

Another possible extension concerning external load would be to categorize the types of external arrivals that may occur, and then to determine their variation during the day. This would result in a more accurate description of external load and this improved accuracy would have a favorable effect on the reliability of simulation model results.

The possibility of assigning priorities within the channel queues is another area worth investigating. The priorities to be assigned to each message type and the priority scheme to be used would need to be determined. The priority scheme could be dynamic as it is in the message queue.

The problem of going from the frequency of arrivals to the distribution of interarrival times needs to be resolved. The development of generalized methods for determining the time between arrivals, given the number of arrivals per time unit, would be desirable.

REFERENCES

[1] De Lutis, Thomas G. and Joseph D. Smith, "IPSS/DBMS: A simulator for modeling data base management systems", Proceedings-1976 Winter Simulation Conference, pp. 526-536.

[2] Edgecomb, Gordon D. and Richard J. Thompson, "Validation - the bottleneck in system simulation", Proceedings-1976 Winter Simulation Conference, pp. 460-466.

[3] IBM Corporation, Information Management System/Virtual Storage (IMS/VS) General Information Manual, Document GH20-1260, IBM Corporation, Data Processing Divsion, White Plains, N.Y.

[4] IBM Corporation, Information Management System/Virtual Storage (IMS/VS) System Application Design Guide, Document SH20-9075, IBM Corporation, Data Processing Division, White Plains, N.Y.

[5] Lavenberg, S. S. and G. S. Shedler, "Stochastic modeling of processor scheduling with application to data base management systems", IBM Journal of Research and Development, 20, pp. 437-448 (September, 1976).

[6] McGee, W. C., "The Information Management System IMS/VS", Parts I through V, IBM Systems Journal, Vol. 16, No. 2, pp. 84-147 (1977).

[7] Pollak, Ted, "A GPSS model of a MVS system", Proceedings-1976 Winter Simulation Conference, pp. 364-371.

[8] Roach, Vincent, "An automated technique for designing optimal performance IMS data bases", FDT-Bulletin of ACM/SIGFIDET, Vol. 6, No. 2, pp. 16-23 (1974).

[9] Rosenshine, Matthew and Stelios Zanakis, "An analysis of computer center operation", AIIE Transactions, Vol. 8, No. 3, pp. 298-306 (September, 1976).

[10] Schneider, Lowell S. and Thomas W. Connolly, "A generalized database system simulator based on the Data-Independent Accessing Model I", Proceedings-1976 Winter Simulation Conference, pp. 513-524.

[11] Tryggestad, Thomas N., "Simulation Techniques for the evaluation of data base systems", Proceedings-1975 Winter Simulation Conference, pp. 468-471.

GLOSSARY

Application Program - User written programs that operate on data contained in the data bases.

Data Base - A nonredundant collection of data items processable by one or more applications. The data base concept allows the integration or sharing of common data.

Limit Count - An attribute of a message type. When the number of messages of this type awaiting processing is greater than or equal the limit count, the priority is raised to the limit priority.

Limit Priority - An attribute of a message type. The normal priority is raised to the limit priority when the number of messages of this type awaiting processing equals or exceeds the limit count.

Message Format Service (MFS) - A comprehensive support facility for various IMS/VS input devices. MFS allows application programs to deal with simple logical messages instead of device dependent data. A program using MFS need not be concerned with the physical characteristics of the I/O device.

Parallel Processing - Allows the same type of application program to process in separate regions.

Preload - Refers to application programs. A preloaded program does not need to be retrieved from the program library.

Processing Limit - The maximum number of messages that a reuseable application program can process before it is removed from processing.

Reuseable - Refers to application programs. A reuseable program can process more than one message each time it is loaded. The maximum number of messages that may be processed depends upon the processing limit.

Serial Processing - The opposite of parallel processing. The same application program cannot execute in more than one region at any given time.

Transaction Code - A MNEMONIC CODE that, when transmitted from a user terminal, initiates IMS processing.

APPENDIX A

THE INPUT GENERATOR

AMOM3	= Vector of third moments.
AVG	= Vector of averages.
FREQ	= Vector containing the number of times each observed frequency occurs.
INFILE	= Counter for arrivals being generated.
IXACT	= Message type being generated.
JCODE	= Current data-code being read. When JCODE = 0, the analysis for the current message type is complete.
KCODE	= Starting data-code (= 033176) for each message type. Indicates that a new message type will be processed.
OBS(J,K)	= Matrix of observed frequencies for message type J in hour K.
PROB	= Vector of cumulative probabilities.
RELF	= Relative frequency.
T	= Arrival time.
VAR	= Vector of variances.

```
C       MAIN PROGRAM
C       READS OBSERVATIONS
      INTEGER XOBS(9,4000),OBS(9,4000)
      COMMON XOBS,AVG(9),VAR(9),AMOM3(9),NDAYS
      COMMON /RAND/ IX,INFILE
      COMMON /TIME/ IXACT
      IX=69697
      INFILE=2
      READ (5,20) NPROB
20    FORMAT (1X,I5)
      NDAYS=66
      DO 120 I=1,NPROB
      IXACT=27
      KCODE=033176
      DO 30 J=1,9
      AMOM3(J)=0
      AVG(J)=0
      VAR(J)=0.0
      DO 30 K=1,90
      OBS(J,K)=0
30    XOBS(J,K)=0
      K=1
C       READ OBSERVED FREQUENCIES
40    READ (5,50) JCODE,(OBS(J,K),J=1,9)
50    FORMAT (1X,I5,8X,9I4)
      IF (JCODE.EQ.0) GO TO 100
      IF (JCODE.NE.KCODE) GO TO 80
60    DO 70 J=1,9
70    XOBS(J,K)=XOBS(J,K)+OBS(J,K)
      GO TO 40
80    K=K+1
      DO 90 J=1,9
```

```
90     OBS(J,K)=OBS(J,K-1)
       KCODE=JCODE
       GO TO 60
100    N=K
       DO 110 J=1,9
       CALL DIST (J,N)
110    CONTINUE
120    CONTINUE
130    FORMAT (I5)
       WRITE (6,130) INFILE
       STOP
       END
```

```
      SUBROUTINE DIST (J,N)
C       DETERMINES CUMULATIVE DISTRIBUTION
      COMMON /RAND/ IX,INFILE
      COMMON XOBS,AVG(9),VAR(9),AMOM3(9),NDAYS
      DIMENSION LIM(300), FREQ(300), RELF(300), PROB(300), G(300), PL(30
     10)
      INTEGER XOBS(9,4000),FREQ
      NWIDE=1
      MAX=0
      MIN=10**9
      DO 10 K=1,N
      IF (XOBS(J,K).LE.MIN) MIN=XOBS(J,K)
      IF (XOBS(J,K).GE.MAX) MAX=XOBS(J,K)
   10 CONTINUE
      NINT=300
      DO 20 L=1,NINT
      RELF(L)=0.0
      PROB(L)=0.0
   20 FREQ(L)=0
      LIM(NINT)=MAX
      MINT=NINT-1
      DO 30 L=1,NINT
   30 LIM(L)=L*NWIDE-1
      DO 50 K=1,N
      DO 40 L=1,NINT
      IF (XOBS(J,K).GT.LIM(L)) GO TO 40
      FREQ(L)=FREQ(L)+1
      GO TO 50
   40 CONTINUE
   50 CONTINUE
C       DETERMINE FREQUENCY
      FREQ(1)=(NDAYS-N)+FREQ(1)
```

```
      CUM=0
      AN=NDAYS
      DO 90 L=1,NINT
C       CALCULATE RELATIVE FREQUENCY
      RELF(L)=FREQ(L)/AN
C       CALCULATE CUMULATIVE PROBABILITY
      CUM=CUM+FREQ(L)
      PROB(L)=CUM/AN
      IF (L.EQ.1) GO TO 70
      IF (PROB(L).GT.PROB(IXMIN).AND.IXMIN.EQ.L-1) GO TO 70
      IF (PROB(L).EQ.PROB(IXMIN)) GO TO 90
      A=L-IXMIN
      B=(PROB(L)-PROB(IXMIN))/A
      IL=IXMIN+1
      IU=L-1
      DO 60 IK=IL,IU
      A=IK-IXMIN
      PROB(IK)=B*A+PROB(IXMIN)
60    WRITE (6,80) LIM(IK),PROB(IK)
70    IXMIN=L
      WRITE (6,80) LIM(L),PROB(L)
80    FORMAT (1X,4HLIM=,I5,2X,9HCUM PROB=,E14.7)
90    CONTINUE
      DO 100 IP=1,300
      IF (PROB(IP).EQ.1.0) GO TO 110
100   CONTINUE
      MMAX=NINT
      GO TO 120
110   MMAX=IP-1
120   CONTINUE
      CALL TIMES (J,MMAX,PROB,LIM)
      RETURN
```

```
      SUBROUTINE TIMES (IHOUR,INDEX,PROB,LIM)
C       GENERATES ARRIVAL TIMES
      COMMON /RAND/ IX,INFILE
      COMMON /TIME/ IXACT
      DIMENSION PROB(300), LIM(300)
      CALL RANDU (IX,IY,YFL)
C       GENERATE UNIFORMLY DISTRIBUTED RANDOM VARIATE ON (0,1)
      IX=IY
      IXL=1
      IXR=INDEX
20    IRAT=(IXR-IXL+1)/2
      IF (IRAT.LE.0) GO TO 30
      MID=IRAT+IXL
C       TABLE LOOK-UP PROCEDURE
      IF (YFL.GT.PROB(MID)) IXL=MID
      IF (YFL.LT.PROB(MID)) IXR=IXR-MID+1
      IF (YFL.EQ.PROB(MID)) GO TO 30
      GO TO 20
C       CORRESPONDING NUMBER OF ARRIVALS FOR THIS HOUR FOUND
30    NXACT=LIM(IXL)
      DO 50 IA=1,NXACT
C       DISTRIBUTE NXACT ARRIVALS UNIFORMLY OVER THIS HOUR
      CALL RANDU (IX,IY,YFL)
      IX=IY
      T=(IHOUR-1)+YFL
      WRITE (6,40) T,IXACT
40    FORMAT (5X,F10.3,I5)
      INFILE=INFILE+1
50    CONTINUE
      RETURN
      END
```

APPENDIX B

THE SIMULATION MODEL

BREG1 = 0 if region 1 is free, 1 otherwise.

BREG2 = 0 if region 2 is free, 1 otherwise.

BUILT1 = End of PSB building in region 1.

BUILT2 = End of PSB building in region 2.

BUSY1 = 0 if channel 1 is free, 1 otherwise.

BUSY2 = 0 if channel 2 is free, 1 otherwise.

CHAN1 = Channel that current data base call for program in region 1 must go to.

CHAN2 = Channel that current data base call for program in region 2 must go to.

CLOCK = Simulator clock. Updated with the passage of each event.

CVALS = Used to accumulated cumulative values during the simulation run.

DBC1 = Current data base being used by program processing in region 1.

DBC2 = Current data base being used by program processing in region 2.

DBTAB = Input matrix corresponding to the channel where each data base is located.

ENDSIM = End of simulation.

ENSCH1 = End of no scheduling for program to process in region 1.

ENSCH2 = End of no scheduling for program to process in region 2.

EOSCH1 = End of scheduling for program to process in region 1.

EOSCH2 = End of scheduling for program to process in region 2.

EPROC1 = End of processing for program in region 1.

EPROC2 = End of processing for program in region 2.

ESERV1 = End of service on channel 1.

ESERV2 = End of service on channel 2.

EVENT = Vector of event times.

INPRO1 = Message processing in region 1.

INPRO2 = Message processing in region 2.

IP1 = Program processing in region 1.

IP2 = Program processing in region 2.

IRCODE = Return code corresponding to region (1 or 2) issuing a data base call.

ITEMP = Temporary array which holds all messages within one class that are awaiting processing.

IX = Starting seed for random number generator.

IXACT = Input matrix of attributes for each message type used. See program code for more detail.

IY = Used for random number generation.

MSGSC1 = Message to process next in region 1.

MSGSC2 = Message to process next in region 2.

MSGT = Accumulates message totals by region.

MSGQ = Vector representing message queue.

MSGQLN = Current length of message queue.

NDBC1 = Number of data base calls for the message processing in region 1.

NDBC2 = Number of data base calls for the message processing in region 2.

NPROG1 = Program processing in region 1.

NPROG2 = Program processing in region 2.

NSERV1 = Message being served on channel 1.

NSERV2 = Message being served on channel 2.

NUM = Counts messages entering the system.

NVIR1 = Number of external transactions generated on channel 1.

NVIR2 = Number of external transactions generated on channel 2.

NXTMS1 = Next message for processing in region 1.

NXTMS2 = Next message for processing in region 2.

PPROC1 = Processing time in region 1 until next data base call.

PPROC2 = Processing time in region 2 until next data base call.

PRELOD = 1 if program is preloaded, 0 otherwise.

PRIOR = Temporary variable containing priority.

PROCH1 = Processing time on channel 1.

PROCH2 = Processing time on channel 2.

PTIME1 = Processing time in region 1.

PTIME2 = Processing time in region 2.

PTYPE1 = Program type processing in region 1.

PTYPE2 = Program type processing in region 2.

QLEN1 = Length of queue at channel 1.

QLEN2 = Length of queue at channel 2.

Q1 = Queue at channel 1.

Q2 = Queue at channel 2.

REUSE	= 1 if program is reuseable, 0 otherwise.
START	= Starting time.
TABLE	= Master table of all transactions in the system and their associated attributes.
TIMARR	= Arrival time of IMS transaction.
TNEXT	= Time of next IMS arrival.
TNVIR1	= Time of next external arrival on channel 1.
TNVIR2	= Time of next external arrival on channel 2.
TOGO1	= Number of data base calls remaining for the message processing in region 1.
TOGO2	= Number of data base calls remaining for the message processing in region 2.

```
C*        MAIN PROGRAM                                                  *
      COMMON /MAIN3/ ONCH1,ONCH2
      COMMON /ENV/ TNVIR1,TNVIR2,NVIR1,NVIR2
      COMMON /RAND/ IX,IY
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /RUSE/ PTYPE1,PTYPE2,TIME1,TIME2
      COMMON /REGN1/ BREG1,INPRO1,EPROC1,PTIME1,DBC1,CHAN1,USE1,NXTMS1,P
     1PROC1,TOGO1,NPROG1,NDBC1
      COMMON /REGN2/ BREG2,INPRO2,EPROC2,PTIME2,DBC2,CHAN2,USE2,NXTMS2,P
     1PROC2,TOGO2,NPROG2,NDBC2
      COMMON /MAIN1/ EOSCH1,ENSCH1,BUILT1,MSGSC1,IRCODE,BUILD1
      COMMON /MAIN2/ EOSCH2,ENSCH2,BUILT2,MSGSC2,BUILD2
      COMMON /XACT/ IXACT
      COMMON /MQUE/ MSGQ,MSGQLN
      COMMON /QUE1/ QLEN1,Q1
      COMMON /QUE2/ QLEN2,Q2
      COMMON /CHANL1/ BUSY1,PROCH1,ESERV1,NSERV1
      COMMON /CHANL2/ BUSY2,PROCH2,ESERV2,NSERV2
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      COMMON /ARRIVE/ TNEXT
      COMMON /DBASE/ DBTAB
      DIMENSION MSGQ(200), TABLE(200,30), Q1(200), Q2(100), EVENT(25)
      DIMENSION IXACT(27,75), DBC1(48), DBC2(48), CVALS(30,10)
      DIMENSION INDQ(200), MSGT(27,3)
      INTEGER QLEN1,QLEN2,Q1,Q2
      INTEGER DBC1,DBC2,PTYPE1,PTYPE2
      INTEGER BREG1,BREG2,CHAN1,CHAN2,USE1,USE2
      INTEGER BUSY1,BUSY2,TOGO1,TOGO2,DBTAB(41,2)
      INTEGER ONCH1,ONCH2
      INTEGER TNVIR1,TNVIR2
      INTEGER TIME1,TIME2
      INTEGER EPROC1,PPROC1,PTIME1
```

```
      INTEGER EPROC2,PPROC2,PTIME2
      INTEGER EOSCH1,ENSCH1,BUILT1,BUILD1
      INTEGER EOSCH2,ENSCH2,BUILT2,BUILD2
      INTEGER PROCH1,PROCH2,ESERV1,ESERV2
      INTEGER TABLE
      DATA TABLE/6000*0/,CVALS/300*0.0/,MSGT/81*0/
C***********************************************************************
C                                                                      *
C                           INITIALIZATION                             *
C                                                                      *
C          TIME UNIT = 1.0E-03 SECONDS     THIS IMPLIES:               *
C                1 SECOND = 1.0E03                                     *
C                1 MINUTE = 6.0E04                                     *
C                1 HOUR = 3.6E06                                       *
C                                                                      *
      START=2.5*3.6E06
      ENDSIM=1.0*3.6E06+START
      IX=67211
      NVIR1=0
      NVIR2=0
      NOREG1=0
      NOREG2=0
      ONCH1=0
      ONCH2=0
      TOGO1=0
      TOGO2=0
      NPROG1=0
      NPROG2=0
      MSGSC1=0
      MSGSC2=0
      TNEXT=START
      TIME1=0
```

```
TIME2=0
INPRO1=0
INPRO2=0
EPROC1=ENDSIM
EPROC2=ENDSIM
PPROC1=ENDSIM
PPROC2=ENDSIM
PTIME1=0
PTIME2=0
CHAN1=0
CHAN2=0
NDBC1=0
NDBC2=0
USE1=0
USE2=0
EOSCH1=ENDSIM
EOSCH2=ENDSIM
ENSCH1=ENDSIM
ENSCH2=ENDSIM
BUILT1=ENDSIM
BUILT2=ENDSIM
TNVIR1=START
TNVIR2=START
TYPE=0
TIMARR=0
PTIME=0
DBCALL=0
TCODE=1
PRIOR=0
REUSE=0
PRELOD=0
DBNUM=0
```

```
      SYSTEM=0
      UPDATE=0
      PROCH1=0
      PROCH2=0
      ESERV1=ENDSIM
      ESERV2=ENDSIM
      NSERV1=0
      NSERV2=0
      PTYPE1=0
      PTYPE2=0
      MSGQLN=0
      QLEN1=0
      NUM=0
      CLOCK=START
      QLEN2=0
      BUSY1=0
      BUSY2=0
      BREG1=0
      BREG2=0
      DO 20 I=1,48
      DBC1(I)=0
      DBC2(I)=0
20    CONTINUE
      DO 30 I=1,200
      Q1(I)=0
      MSGQ(I)=0
30    CONTINUE
      DO 40 I=1,100
      Q2(I)=0
40    CONTINUE
C***********************************************************************
C                                                                      *
```

```
C        IXACT(I,1)=TRANSACTION TYPE                                          *
C        IXACT(I,2)=PROCESSING TIME                                           *
C        IXACT(I,3)=PRIORITY IN REGION 1                                      *
C        IXACT(I,4)=PRIORITY IN REGION 2                                      *
C        IXACT(I,5)=PRIORITY IN REGION 3                                      *
C        IXACT(I,6)=1 IF REUSABLE, 0 OTHERWISE                                *
C        IXACT(I,7)=1 IF PRELOADED, 0 OTHERWISE                               *
C        IXACT(I,8)=PROCESSING LIMIT                                          *
C        IXACT(I,9)=LIMIT PRIORITY                                            *
C        IXACT(I,10)=LIMIT COUNT
C        IXACT(I,11)=PROGRAM NUMBER
C        IXACT(I,12)=CLASS                                                    *
C        IXACT(I,13)=1 IF SERIAL, 0 OTHERWISE                                 *
C        IXACT(I,14)=CHANNEL TIME                                             *
C        IXACT(I,15)=NUMBER OF DB CALLS                                       *
C        IXACT(I,16) TO IXACT(I,75)=DB LIST                                   *
C                                                                             *
C******************************************************************************
      DO 80 I=1,27
      READ (5,50) (IXACT(I,J),J=1,15)
50    FORMAT (15I5)
      IEN=IXACT(I,15)+15
      READ (5,60) (IXACT(I,J),J=16,IEN)
60    FORMAT (40I2)
      IST=IEN+1
      DO 70 J=IST,75
70    IXACT(I,J)=0
80    CONTINUE
      DO 100 I=1,41
      READ (5,90) (DBTAB(I,J),J=1,2)
90    FORMAT (2I2)
100   CONTINUE
```

```
      DO 120 I=1,100000
      READ (9,110) TIME
110   FORMAT (F10.5)
      IF (TIME.GE.(START/3.6E06)) GO TO 130
120   CONTINUE
C                                                                      *
C                                                                      *
C***********************************************************************
130   CONTINUE
140   CONTINUE
C*       NEXT EVENT ROUTINE                                            *
      EVENT(1)=ENDSIM
      EVENT(2)=ESERV1
      EVENT(3)=ESERV2
      EVENT(4)=EOSCH1
      EVENT(5)=EOSCH2
      EVENT(6)=ENSCH1
      EVENT(7)=ENSCH2
      EVENT(8)=BUILT1
      EVENT(9)=BUILT2
      EVENT(10)=PPROC1
      EVENT(11)=PPROC2
      EVENT(12)=EPROC1
      EVENT(13)=EPROC2
      EVENT(14)=TNEXT
      EVENT(15)=TNVIR1
      EVENT(16)=TNVIR2
      XMIN=EVENT(1)
      KEEP=1
      DO 150 IA=1,16
      IF (EVENT(IA).GE.XMIN) GO TO 150
      XMIN=EVENT(IA)
```

```
      KEEP=IA
150   CONTINUE
      DO 200 IC=1,27
      INQ=0
      INDQL=0
      IF (MSGQLN.EQ.0) GO TO 200
      DO 160 IB=1,MSGQLN
      JA=LOCATE(MSGQ(IB))
      IF (IC.NE.TABLE(JA,2)) GO TO 160
      INQ=INQ+1
      INDQL=INDQL+1
      INDQ(INDQL)=JA
160   CONTINUE
      IF (INQ.EQ.0) GO TO 200
      IF (INQ.GE.TABLE(JA,15)) GO TO 180
      DO 170 JC=1,INDQL
      TABLE(INDQ(JC),8)=TABLE(JA,23)
      TABLE(INDQ(JC),21)=TABLE(JA,24)
      TABLE(INDQ(JC),22)=TABLE(JA,25)
170   CONTINUE
      GO TO 200
180   DO 190 JC=1,INDQL
      TABLE(INDQ(JC),8)=TABLE(JA,14)
      TABLE(INDQ(JC),21)=TABLE(JA,14)
      TABLE(INDQ(JC),22)=TABLE(JA,14)
190   CONTINUE
200   CONTINUE
C*         NEXT EVENT DETERMINED                                            *
C*         MAKE APPROPRIATE CALLS                                           *
      GO TO (310,210,220,230,240,250,260,270,280,320,340,290,300,360,370
     1,380), KEEP
210   ONCH1=ENDSIM*10.
```

```
      CALL FREE1
      ESERV1=CLOCK+ONCH1
      GO TO 390
220   ONCH2=ENDSIM*10.
      CALL FREE2
      ESERV2=CLOCK+ONCH2
      GO TO 390
230   SCHD=ENDSIM*10.
      CLOCK=EOSCH1
      CALL REG1 (3)
      EOSCH1=CLOCK+SCHD
      GO TO 390
240   SCHD=ENDSIM*10.
      CLOCK=EOSCH2
      CALL REG2 (3)
      EOSCH2=CLOCK+SCHD
      GO TO 390
250   CLOCK=ENSCH1
      ENSCH1=ENDSIM*10.
      CALL REG1 (3)
      GO TO 390
260   CLOCK=ENSCH2
      ENSCH2=ENDSIM*10.
      CALL REG2 (3)
      GO TO 390
270   BUILD1=ENDSIM*10.
      CLOCK=BUILT1
      CALL REG1 (4)
      BUILT1=BUILD1+CLOCK
      GO TO 390
280   BUILD2=ENDSIM*10.
      CLOCK=BUILT2
```

```
      CALL REG2 (4)
      BUILT2=BUILD2+CLOCK
      GO TO 390
290   CALL REG1 (6)
      GO TO 390
300   CALL REG2 (6)
      GO TO 390
310   CALL STATS
      STOP
320   CLOCK=PPROC1
      IF (NDBC1.LT.1) GO TO 330
      IF (CHAN1.EQ.1) CALL CHNL1 (INPRO1)
      IF (CHAN1.EQ.2) CALL CHNL2 (INPRO1)
      NDBC1=NDBC1-1
330   PPROC1=ENDSIM*10.
      GO TO 390
340   CLOCK=PPROC2
      IF (NDBC2.LT.1) GO TO 350
      IF (CHAN2.EQ.1) CALL CHNL1 (INPRO2)
      IF (CHAN2.EQ.2) CALL CHNL2 (INPRO2)
      NDBC2=NDBC2-1
350   PPROC2=ENDSIM*10.
      GO TO 390
360   CALL ARIVAL
      GO TO 390
370   CLOCK=TNVIR1
      CALL ENVIR1
      GO TO 390
380   CLOCK=TNVIR2
      CALL ENVIR2
390   CONTINUE
      IF (MSGQLN.EQ.0) GO TO 140
```

```
IF (BREG1.EQ.0) CALL REG1 (1)
IF (BREG2.EQ.0) CALL REG2 (1)
GO TO 140
END
```

```
      SUBROUTINE FREE1
C*       FREES CHANNEL1 WHEN PROCESSING IS COMPLETED                   *
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /REGN2/ BREG2,INPRO2,EPROC2,PTIME2,DBC2,CHAN2,USE2,NXTMS2,P
     1PROC2,TOGO2,NPROG2,NDBC2
      COMMON /REGN1/ BREG1,INPRO1,EPROC1,PTIME1,DBC1,CHAN1,USE1,NXTMS1,P
     1PROC1,TOGO1,NPROG1,NDBC1
      COMMON /CHANL1/ BUSY1,PROCH1,ESERV1,NSERV1
      COMMON /QUE1/ QLEN1,Q1
      COMMON /MQUE/ MSGQ,MSGQLN
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      COMMON /MAIN1/ EOSCH1,ENSCH1,BUILT1,MSGSC1,IRCODE,BUILD1
      COMMON /MAIN2/ EOSCH2,ENSCH2,BUILT2,MSGSC2,BUILD2
      DIMENSION TABLE(200,30), Q1(200), DBC1(48), DBC2(48), CVALS(30,10)
      DIMENSION MSGQ(200)
      INTEGER QLEN1,Q1
      INTEGER BREG1,BREG2,BUSY1
      INTEGER EPROC2,PTIME2,DBC2,CHAN2,USE2,PPROC2,TOGO2
      INTEGER EPROC1,PTIME1,DBC1,CHAN1,USE1,PPROC1,TOGO1
      INTEGER PROCH1,ESERV1
      INTEGER TABLE
      INTEGER EOSCH1,ENSCH1,BUILT1,BUILD1
      INTEGER EOSCH2,ENSCH2,BUILT2,BUILD2
      BUSY1=0
      I=LOCATE(NSERV1)
      JA=I
      JGO=TABLE(I,7)
C***********************************************************************
C                             JGO=1: MSG TO BE PROCESSED               *
C                             JGO=2: CALL TO CHANNEL 1 OR NO SCHEDULING *
C                             JGO=3: DETERMINE END OF SCHEDULING       *
C                             JGO=4: BLOCK BUILDING                    *
```

```
C                                JGO=6: DB CALL FROM REGION                              *
C                                JGO=7: ENVIRONMENT                                      *
C*****************************************************************************************
      IREG=TABLE(I,6)
      CLOCK=ESERV1
      TINSC1=CLOCK-TABLE(I,11)
      IND=TABLE(I,2)
      CVALS(IND,3)=CVALS(IND,3)+TINSC1
      TABLE(I,11)=CLOCK
      GO TO (10,30,50,60,90,100,110), JGO
10    MSGQLN=MSGQLN+1
      IF (MSGQLN.GT.200) WRITE (6,20) MSGQLN
      MSGQ(MSGQLN)=NSERV1
      TABLE(JA,20)=2
20    FORMAT (5X,26H*** MESSAGE QUEUE LENGTH =,I5,3H***)
      TABLE(I,7)=2
30    IF (BREG1.EQ.1) GO TO 40
      GO TO 130
40    IF (BREG2.EQ.1) GO TO 120
      GO TO 140
50    IF (IREG.EQ.1) CALL REG1 (2)
      IF (IREG.EQ.2) CALL REG2 (2)
      GO TO 120
60    CONTINUE
      GO TO (70,80), IREG
70    CALL REG1 (3)
      TABLE(I,7)=5
      GO TO 120
80    CALL REG2 (3)
      TABLE(I,7)=5
      GO TO 120
```

```
90     IF (IREG.EQ.1) CALL REG1 (4)
       IF (IREG.EQ.2) CALL REG2 (4)
       TABLE(I,7)=6
       GO TO 120
100    IF (IREG.EQ.1) CALL REG1 (5)
       IF (IREG.EQ.2) CALL REG2 (5)
       GO TO 120
110    TABLE(I,20)=0
120    IF (QLEN1.EQ.0) RETURN
       IF (BUSY1.EQ.1) RETURN
       CALL CHNL1 (-1)
       RETURN
130    CALL REG1 (1)
       TABLE(I,6)=1
       GO TO 120
140    CALL REG2 (1)
       TABLE(I,6)=2
       GO TO 120
       END
```

```
      SUBROUTINE FREE2
C*       FREES CHANNEL2 WHEN PROCESSING IS COMPLETED                    *
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /REGN1/ BREG1,INPRO1,EPROC1,PTIME1,DBC1,CHAN1,USE1,NXTMS1,P
     1PROC1,TOGO1,NPROG1,NDBC1
      COMMON /REGN2/ BREG2,INPRO2,EPROC2,PTIME2,DBC2,CHAN2,USE2,NXTMS2,P
     1PROC2,TOGO2,NPROG2,NDBC2
      COMMON /CHANL2/ BUSY2,PROCH2,ESERV2,NSERV2
      COMMON /QUE2/ QLEN2,Q2
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      COMMON /MAIN1/ EOSCH1,ENSCH1,BUILT1,MSGSC1,IRCODE,BUILD1
      COMMON /MAIN2/ EOSCH2,ENSCH2,BUILT2,MSGSC2,BUILD2
      DIMENSION TABLE(200,30), Q2(100), DBC1(48), DBC2(48)
      DIMENSION CVALS(30,10)
      INTEGER QLEN2,Q2,BUSY2
      INTEGER BREG1,EPROC1,PTIME1,DBC1,CHAN1,USE1,PPROC1,TOGO1
      INTEGER BREG2,EPROC2,PTIME2,DBC2,CHAN2,USE2,PPROC2,TOGO2
      INTEGER PROCH2,ESERV2
      INTEGER TABLE
      INTEGER EOSCH1,ENSCH1,BUILT1,BUILD1
      INTEGER EOSCH2,ENSCH2,BULIT2,BUILD2
      BUSY2=0
      I=LOCATE(NSERV2)
      JGO=TABLE(I,7)
C***********************************************************************
C                           JGO=5: START PROCESSING                    *
C                           JGO=6: DB CALL FROM REGION                 *
C                           JGO=7: ENVIRONMENT                         *
C***********************************************************************
      IREG=TABLE(I,6)
      CLOCK=ESERV2
      TINSC2=CLOCK-TABLE(I,11)
```

```
      IND=TABLE(I,2)
      CVALS(IND,5)=CVALS(IND,5)+TINSC2
      TABLE(I,11)=CLOCK
      GO TO (60,60,60,60,10,40,50), JGO
10    CONTINUE
      GO TO (20,30), IREG
20    CALL REG1 (4)
      TABLE(I,7)=6
      GO TO 60
30    CALL REG2 (4)
      TABLE(I,7)=6
      GO TO 60
40    IF (IREG.EQ.1) CALL REG1 (5)
      IF (IREG.EQ.2) CALL REG2 (5)
      GO TO 60
50    TABLE(I,20)=0
60    IF (QLEN2.EQ.0) RETURN
      IF (BUSY2.EQ.1) RETURN
      CALL CHNL2 (-1)
      RETURN
      END
```

```
      SUBROUTINE CHNL1 (IPM)
C*       SIMULATES OPERATION OF CHANNEL1                                   *
      COMMON /MAIN3/ ONCH1,ONCH2
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /RAND/ IX,IY
      COMMON /CHANL1/ BUSY1,PROCH1,ESERV1,NSERV1
      COMMON /QUE1/ QLEN1,Q1
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      DIMENSION TABLE(200,30), CVALS(30,10)
      INTEGER QLEN1,Q1(200),BUSY1
      INTEGER ONCH1,ONCH2
      INTEGER PROCH1,ESERV1
      INTEGER TABLE
      IF (IPM) 30,20,10
C***********************************************************************
C                   IPM < 0 IMPLIES A CALL FROM FREE1                  *
C                   IPM = 0 IMPLIES A CALL FROM ARIVAL                 *
C                   IPM > 0 IMPLIES OTHER CALLS                        *
C***********************************************************************
10    QLEN1=QLEN1+1
      Q1(QLEN1)=IPM
      JA=LOCATE(IPM)
      TABLE(JA,11)=CLOCK
20    IF (BUSY1.EQ.1) RETURN
30    NSERV1=Q1(1)
      JA=LOCATE(NSERV1)
      PROCH1=TABLE(JA,19)/TABLE(JA,5)
      ONCH1=PROCH1
      TINQ1=CLOCK-TABLE(JA,11)
      IND=TABLE(JA,2)
      CVALS(IND,2)=CVALS(IND,2)+TINQ1
      CVALS(IND,9)=CVALS(IND,9)+1
```

```
      TABLE(JA,11)=CLOCK
      ESERV1=CLOCK+ONCH1
      BUSY1=1
      QLEN1=QLEN1-1
      DO 40 I=1,199
      Q1(I)=Q1(I+1)
40    CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CHNL2 (IPM)
C*         SIMULATES OPERATION OF CHANNEL2                              *
      COMMON /MAIN3/ ONCH1,ONCH2
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /RAND/ IX,IY
      COMMON /CHANL2/ BUSY2,PROCH2,ESERV2,NSERV2
      COMMON /QUE2/ QLEN2,Q2
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      DIMENSION TABLE(200,30), CVALS(30,10)
      INTEGER QLEN2,Q2(100),BUSY2
      INTEGER ONCH1,ONCH2
      INTEGER PROCH2,ESERV2,TABLE
      IF (IPM) 20,10,10
C***********************************************************************
C                     IPM < 0 IMPLIES A CALL FROM FREE2                *
C***********************************************************************
10    QLEN2=QLEN2+1
      Q2(QLEN2)=IPM
      JA=LOCATE(IPM)
      TABLE(JA,11)=CLOCK
      IF (BUSY2.EQ.1) RETURN
20    NSERV2=Q2(1)
      JA=LOCATE(NSERV2)
      PROCH2=TABLE(JA,19)/TABLE(JA,5)
      ONCH2=PROCH2
      TINQ2=CLOCK-TABLE(JA,11)
      IND=TABLE(JA,2)
      CVALS(IND,4)=CVALS(IND,4)+TINQ2
      CVALS(IND,10)=CVALS(IND,10)+1
      TABLE(JA,11)=CLOCK
      QLEN2=QLEN2-1
      DO 30 I=1,99
```

```
      Q2(I)=Q2(I+1)
30    CONTINUE
      ESERV2=CLOCK+ONCH2
      BUSY2=1
      RETURN
      END
```

```
      SUBROUTINE NXTMSG (IREG)
C*       SELECTS NEXT MESSAGE FOR PROCESSING                                    *
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /MAIN1/ EOSCH1,ENSCH1,BUILT1,MSGSC1,IRCODE,BUILD1
      COMMON /MAIN2/ EOSCH2,ENSCH2,BUILT2,MSGSC2,BUILD2
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      COMMON /MQUE/ MSGQ,MSGQLN
      COMMON /REGN1/ BREG1,INPRO1,EPROC1,PTIME1,DBC1,CHAN1,USE1,NXTMS1,P
     1PROC1,TOGO1,NPROG1,NDBC1
      COMMON /REGN2/ BREG2,INPRO2,EPROC2,PTIME2,DBC2,CHAN2,USE2,NXTMS2,P
     1PROC2,TOGO2,NPROG2,NDBC2
      DIMENSION TABLE(200,30), MSGQ(200), IP1(3), IP2(3), MSGT(27,3)
      DIMENSION ITEMP(200,5), DBC1(48), DBC2(48), CVALS(30,10)
      INTEGER EOSCH1,ENSCH1,BUILT1,BUILD1
      INTEGER EOSCH2,ENSCH2,BUILT2,BUILD2
      INTEGER TABLE
      INTEGER BREG1,EPROC1,PTIME1,DBC1,CHAN1,USE1,PPROC1,TOGO1
      INTEGER BREG2,EPROC2,PTIME2,DBC2,CHAN2,USE2,PPROC2,TOGO2
      DATA IP1/3,2,1/,IP2/3,1,2/
      GO TO (10,100), IREG
10    IF (MSGQLN.EQ.0) GO TO 160
      DO 40 J=1,3
      KX=0
      DO 30 I=1,MSGQLN
      JA=LOCATE(MSGQ(I))
      IF (TABLE(JA,17).EQ.IP1(J).AND.TABLE(JA,18).EQ.0) GO TO 20
      IF (TABLE(JA,17).EQ.IP1(J).AND.TABLE(JA,18).EQ.1.AND.TABLE(JA,16).
     1NE.NPROG2) GO TO 20
      GO TO 30
20    KX=KX+1
      ITEMP(KX,1)=MSGQ(I)
      ITEMP(KX,2)=TABLE(JA,8)
```

```
30    CONTINUE
      IF (KX.EQ.0) GO TO 40
      GO TO 50
40    CONTINUE
      GO TO 160
C***********************************************************************
C                     SORT ITEMP ON COLUMN 2 (PRIORITY)               *
C***********************************************************************
50    CONTINUE
      CALL SHELL (ITEMP,2,KX,2)
      JA=LOCATE(ITEMP(1,1))
      NXTMS1=TABLE(JA,1)
      MSGSC1=NXTMS1
      NOREG1=NOREG1+1
      DO 60 JQ=1,MSGQLN
      IF (MSGQ(JQ).EQ.NXTMS1) GO TO 80
60    CONTINUE
      WRITE (2,70) NXTMS1
70    FORMAT (5X,3H***,I10,36HCANNOT BE FOUND IN MESSAGE QUEUE.***)
      GO TO 160
80    DO 90 KK=JQ,199
      MSGQ(KK)=MSGQ(KK+1)
90    CONTINUE
      MSGQLN=MSGQLN-1
      TIMSGQ=CLOCK-TABLE(JA,11)
      IND=TABLE(JA,2)
      CVALS(IND,6)=CVALS(IND,6)+TIMSGQ
      TABLE(JA,11)=CLOCK
      MSGT(TABLE(JA,2),IREG)=MSGT(TABLE(JA,2),IREG)+1
      RETURN
100   IF (MSGQLN.EQ.0) GO TO 170
      DO 150 J=1,3
```

```
      KY=0
      DO 120 I=1,MSGQLN
      JA=LOCATE(MSGQ(I))
      IF (TABLE(JA,17).EQ.IP2(J).AND.TABLE(JA,18).EQ.0) GO TO 110
      IF (TABLE(JA,17).EQ.IP2(J).AND.TABLE(JA,18).EQ.1.AND.TABLE(JA,16).
     1NE.NPROG1) GO TO 110
      GO TO 120
110   KY=KY+1
      ITEMP(KY,1)=MSGQ(I)
      ITEMP(KY,2)=TABLE(JA,21)
120   CONTINUE
      IF (KY.EQ.0) GO TO 150
C***********************************************************************
C                        SORT ITEMP ON COLUMN 2 (PRIORITY)             *
C***********************************************************************
      CALL SHELL (ITEMP,2,KY,2)
      JA=LOCATE(ITEMP(1,1))
      NXTMS2=TABLE(JA,1)
      DO 130 JQ=1,MSGQLN
      IF (MSGQ(JQ).EQ.NXTMS2) GO TO 140
130   CONTINUE
140   CONTINUE
      MSGSC2=NXTMS2
      NOREG2=NOREG2+1
      GO TO 80
150   CONTINUE
      GO TO 170
160   NXTMS1=0
      MSGSC1=0
      RETURN
170   NXTMS2=0
      MSGSC2=0
```

```
RETURN
END
```

```
      SUBROUTINE ARIVAL
C*      GENERATES ARRIVALS                                              *
      COMMON /CHANL1/ BUSY1,PROCH1,ESERV1,NSERV1
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /RAND/ IX,IY
      COMMON /QUE1/ QLEN1,Q1
      COMMON /XACT/ IXACT
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      COMMON /ARRIVE/ TNEXT
      DIMENSION IXACT(27,75), TABLE(200,30), CVALS(30,10)
      INTEGER QLEN1,Q1(200)
      INTEGER BUSY1,PROCH1,ESERV1
      INTEGER TABLE
      NUM=NUM+1
      READ (9,10,END=70) TIME,ITYPE
10    FORMAT (F10.5,I5)
      R=.34
      TIME=R*TIME*3.6E06+(1.0-R)*START
      TYPE=ITYPE
      TCODE=1
      TIMARR=TNEXT
      CLOCK=TIMARR
      TNEXT=TIME
      DO 20 I=1,200
      IF (TABLE(I,20).EQ.0) GO TO 40
20    CONTINUE
      WRITE (6,30)
30    FORMAT (1X,///,5X,20H***  TABLE FULL  ***,///)
      CALL STATS
      STOP
40    TABLE(I,1)=NUM
      TABLE(I,2)=IXACT(ITYPE,1)
```

```
      TABLE(I,3)=TIMARR
      TABLE(I,4)=IXACT(ITYPE,2)
      TABLE(I,5)=IXACT(ITYPE,15)
      TABLE(I,6)=0
      TABLE(I,7)=1
      TABLE(I,8)=IXACT(ITYPE,3)
      TABLE(I,9)=IXACT(ITYPE,6)
      TABLE(I,10)=IXACT(ITYPE,7)
      TABLE(I,11)=CLOCK
      TABLE(I,12)=1
      TABLE(I,13)=IXACT(ITYPE,8)
      TABLE(I,14)=IXACT(ITYPE,9)
      TABLE(I,15)=IXACT(ITYPE,10)
      TABLE(I,16)=IXACT(ITYPE,11)
      TABLE(I,17)=IXACT(ITYPE,12)
      TABLE(I,18)=IXACT(ITYPE,13)
      TABLE(I,19)=IXACT(ITYPE,14)
      TABLE(I,20)=1
      TABLE(I,21)=IXACT(ITYPE,4)
      TABLE(I,22)=IXACT(ITYPE,5)
      TABLE(I,23)=IXACT(ITYPE,3)
      TABLE(I,24)=IXACT(ITYPE,4)
      TABLE(I,25)=IXACT(ITYPE,5)
      IF (QLEN1.EQ.0) GO TO 60
      DO 50 IJ=1,199
      IK=201-IJ
      Q1(IK)=Q1(IK-1)
50    CONTINUE
60    Q1(1)=NUM
      QLEN1=QLEN1+1
      CVALS(ITYPE,1)=CVALS(ITYPE,1)+1
      PROCH1=10
```

```
      CALL CHNL1 (0)
      RETURN
70    CLOCK=ENDSIM
      TNEXT=ENDSIM*10.
      RETURN
      END
```

```
      SUBROUTINE STATS
C*       CALCULATES STATISTICS FROM DATA ACCUMULATED                      *
      COMMON /REGN1/ BREG1,INPRO1,EPROC1,PTIME1,DBC1,CHAN1,USE1,NXTMS1,P
     1PROC1,TOGO1,NPROG1,NDBC1
      COMMON /REGN2/ BREG2,INPRO2,EPROC2,PTIME2,DBC2,CHAN2,USE2,NXTMS2,P
     1PROC2,TOGO2,NPROG2,NDBC2
      COMMON /MQUE/ MSGQ,MSGQLN
      COMMON /QUE1/ QLEN1,Q1
      COMMON /QUE2/ QLEN2,Q2
      COMMON /CHANL1/ BUSY1,PROCH1,ESERV1,NSERV1
      COMMON /CHANL2/ BUSY2,PROCH2,ESERV2,NSERV2
      COMMON /ENV/ TNVIR1,TNVIR2,NVIR1,NVIR2
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      INTEGER BUSY1,BUSY2,TOGO1,TOGO2,BREG1,BREG2,CHAN1,CHAN2
      INTEGER USE1,USE2,DBC1(48),DBC2(48),QLEN1,QLEN2,Q1(200),Q2(100)
      INTEGER EPROC1,PTIME1,PPROC1
      INTEGER EPROC2,PTIME2,PPROC2
      INTEGER PROCH1,PROCH2,ESERV1,ESERV2
      INTEGER TNVIR1,TNVIR2
      INTEGER TABLE
      DIMENSION MSGQ(200), MSGT(27,3), ACTUAL(27)
      DIMENSION TABLE(200,30), CVALS(30,10)
      DATA ACTUAL/1.15,3.50,1.62,1.03,2.21,1.13,0.84,0.95,1.01,1.17,1.08
     1,1.04,1.00,.83,2.13,.98,1.38,0.56,1.09,1.27,1.52,2.29,1.95,1.93,1.
     253,1.02,0.0/
C*************************************************************************
C                                                                        *
C                              STATISTICS                                *
C                                                                        *
C*************************************************************************
      NPRO=NUM-1
```

```
      WRITE (6,10) NPRO
10    FORMAT (5X,I5,29HMESSAGES HAVE BEEN PROCESSED.)
      WRITE (6,20) CLOCK
20    FORMAT (5X,28HTIME AT END OF SIMULATION = ,F20.2)
      DO 40 J=1,28
      DO 30 I=2,10
      IF (CVALS(J,1).EQ.0.0) GO TO 30
      CVALS(J,I)=CVALS(J,I)/CVALS(J,1)
30    CONTINUE
40    CONTINUE
      WRITE (6,50)
50    FORMAT (1H1,5X,44H*****   AVERAGES BY TRANSACTION CODE   *****//)
      DO 110 I=1,27
      IF (CVALS(I,9).EQ.0.0) GO TO 60
      ATC1=CVALS(I,3)/CVALS(I,9)
      GO TO 70
60    ATC1=0.0
70    IF (CVALS(I,10).EQ.0.0) GO TO 80
      ATC2=CVALS(I,5)/CVALS(I,10)
      GO TO 90
80    ATC2=0.0
90    WRITE (6,100) I,(CVALS(I,J),J=1,10),ATC1,ATC2
100   FORMAT (5X,I3,10E9.5,2E9.5)
110   CONTINUE
      WRITE (6,120) NOREG1
120   FORMAT (10X,9HREGION 1:,I5)
      WRITE (6,130) NOREG2
130   FORMAT (10X,9HREGION 2:,I5)
      WRITE (6,140)
140   FORMAT (1H1,5X,36H*****   VALUES AT END OF RUN   *****,///)
      WRITE (6,150) MSGQLN
150   FORMAT (5X,22HMESSAGE QUEUE LENGTH =,I5)
```

```
      WRITE (6,160) QLEN1
160   FORMAT (5X,24HCHANNEL 1 QUEUE LENGTH =,I5)
      WRITE (6,170) QLEN2
170   FORMAT (5X,24HCHANNEL 2 QUEUE LENGTH =,I5)
      WRITE (6,180) NSERV1
180   FORMAT (5X,24HPROCESSING ON CHANNEL 1:,I5)
      WRITE (6,190) NSERV2
190   FORMAT (5X,24HPROCESSING ON CHANNEL 2:,I5)
      WRITE (6,200) INPRO1
200   FORMAT (5X,23HPROCESSING IN REGION 1:,I5)
      WRITE (6,210) INPRO2
210   FORMAT (5X,23HPROCESSING IN REGION 2:,I5)
      SUM=0.0
      RSUM=0.0
      DO 220 I=1,27
220   CONTINUE
      ARESP=$TOTAL/$XACT
      WRITE (6,230) $XACT
230   FORMAT (5X,///,20H TOTAL TRANSACTIONS:,F15.0)
      WRITE (6,240) ARESP
240   FORMAT (5X,31HAVERAGE OVERALL RESPONSE TIME= ,E12.4,13H MILLISECON
     1DS)
      WRITE (6,250)
250   FORMAT (1H1,5X,30H***** MESSAGES BY REGION *****,///)
      WRITE (6,260)
260   FORMAT (10X,4HXACT,10X,8HREGION 1,5X,8HREGION 2,5X,8HREGION 3)
      DO 280 II=1,27
      WRITE (6,270) II,(MSGT(II,IJ),IJ=1,3)
270   FORMAT (12X,I2,13X,I4,9X,I4,9X,I4)
280   CONTINUE
      WRITE (6,290)
290   FORMAT (1H1,5X,39H***** VALIDATION OF RESPONSE TIME *****,///)
```

```
      WRITE (6,300)
300   FORMAT (1H0,9X,4HXACT,10X,6HACTUAL,10X,9HSIMULATED,10X,5H A/S ,10X
     1,9HFREQUENCY,/)
      DO 320 IT=1,26
      SIM=CVALS(IT,8)/1000.
      IF (SIM.EQ.0.0) GO TO 320
      AOS=ACTUAL(IT)/SIM
      SUM=SUM+CVALS(IT,1)*(AOS-1.0)**2
      RSUM=RSUM+CVALS(IT,1)*AOS
      WRITE (6,310) IT,ACTUAL(IT),SIM,AOS,CVALS(IT,1)
310   FORMAT (10X,I2,10X,F6.2,10X,F9.2,10X,F10.4,10X,F9.0)
320   CONTINUE
      WRITE (6,330) $XACT
330   FORMAT (77X,F10.0)
      SSUM=(SUM/$XACT)**.5
      RBAR=RSUM/$XACT
      WRITE (6,340) RBAR,SSUM
340   FORMAT (5X,8HR-BAR = ,F8.5,5X,5H S = ,F8.5)
      RETURN
      END
```

```
      SUBROUTINE REG1 (ITR)
C*       SIMULATES OPERATION OF REGION1                                  *
      COMMON /CHANL1/ BUSY1,PROCH1,ESERV1,NSERV1
      COMMON /CHANL2/ BUSY2,PROCH2,ESERV2,NSERV2
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /RAND/ IX,IY
      COMMON /XACT/ IXACT
      COMMON /REGN1/ BREG1,INPRO1,EPROC1,PTIME1,DBC1,CHAN1,USE1,NXTMS1,P
     1PROC1,TOGO1,NPROG1,NDBC1
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      COMMON /RUSE/ PTYPE1,PTYPE2,TIME1,TIME2
      COMMON /MAIN1/ EOSCH1,ENSCH1,BUILT1,MSGSC1, IRCODE,BUILD1
      COMMON /MAIN2/ EOSCH2,ENSCH2,BUILT2,MSGSC2,BUILD2
      COMMON /DBASE/ DBTAB
      INTEGER DBC1,PTYPE1,PTYPE2
      INTEGER CHAN1,USE1,BREG1,TOGO1,DBTAB(41,2)
      INTEGER BUSY1,PROCH1,ESERV1,BUSY2,PROCH2,ESERV2
      INTEGER EPROC1,PTIME1,PPROC1
      INTEGER TABLE
      INTEGER EOSCH1,EOSCH2,ENSCH1,ENSCH2,BUILT1,BUILT2,BUILD1,BUILD2
      INTEGER TIME1,TIME2
      DIMENSION TABLE(200,30), DBC1(48), IXACT(27,75), CVALS(30,10)
      IRCODE=1
      GO TO (10,30,40,60,80,120), ITR
10    CALL NXTMSG (1)
      IF (MSGSC1.EQ.0) RETURN
      JA=LOCATE(MSGSC1)
      TABLE(JA,6)=1
      NPROG1=TABLE(JA,16)
      TOGO1=TABLE(JA,13)
      NXTMS1=MSGSC1
      IF (TABLE(JA,10).EQ.0) GO TO 20
```

```
      ENSCH1=CLOCK
      TABLE(JA,7)=4
      BREG1=1
      RETURN
20    CALL CHNL1 (MSGSC1)
      TABLE(JA,7)=3
      BREG1=1
      RETURN
30    EOSCH1=CLOCK+10
      BREG1=1
      JA=LOCATE(MSGSC1)
      TABLE(JA,7)=4
      RETURN
40    JA=LOCATE(NXTMS1)
      TABLE(JA,7)=5
      BREG1=1
      IF (TABLE(JA,10).EQ.0) GO TO 50
      BUILT1=CLOCK
      RETURN
50    CONTINUE
      BUILD1=10
      BUILT1=BUILD1+CLOCK
      RETURN
60    BREG1=1
      INPRO1=NXTMS1
      JA=LOCATE(INPRO1)
      TABLE(JA,7)=6
      NDBC1=TABLE(JA,5)
      INDX=TABLE(JA,2)
      USE1=TABLE(JA,9)
      PTYPE1=TABLE(JA,16)
      PTIME1=TABLE(JA,4)
```

```
      DO 70 ID=1,47
      DBC1(ID)=IXACT(INDX,ID+15)
70    CONTINUE
80    CONTINUE
      IF (NDBC1.LE.0) GO TO 110
      IDB=DBC1(NDBC1)
      CHAN1=DBTAB(IDB,2)
C***********************************************************************
C                         GENERATE PARTIAL PROCESSING TIME              *
C***********************************************************************
      CALL RNDNO (IX,IY,PT)
      IX=IY
      JA=LOCATE(INPRO1)
      PPROC1=PT*PTIME1
      IND=TABLE(JA,2)
      CVALS(IND,7)=CVALS(IND,7)+PPROC1
      PTIME1=PTIME1-PPROC1
      PPROC1=CLOCK+PPROC1
      IF (PTIME1.LE.0.0) GO TO 90
      IF (NDBC1.LE.0) GO TO 110
      RETURN
90    CONTINUE
      IF (NDBC1.LE.0) GO TO 110
      DO 100 JQ=1,NDBC1
      IF (CHAN1.EQ.1) CALL CHNL1 (INPRO1)
      IF (CHAN1.EQ.2) CALL CHNL2 (INPRO1)
100   CONTINUE
      NDBC1=0
      RETURN
110   EPROC1=CLOCK+PTIME1
      PPROC1=ENDSIM*10.
      TABLE(JA,5)=0
```

```
      RETURN
120   BREG1=0
      CLOCK=EPROC1
      EPROC1=ENDSIM*10.
      JA=LOCATE(INPRO1)
      TINSYS=CLOCK-TABLE(JA,3)
      IND=TABLE(JA,2)
      CVALS(IND,8)=CVALS(IND,8)+TINSYS
      TABLE(JA,20)=0
      INPRO1=0
      NPROG1=0
      IF (TOGO1.LE.0.OR.TABLE(JA,13).LE.1) GO TO 10
      CALL REUSE (1)
      TOGO1=TOGO1-1
      IF (NXTMS1.EQ.0) GO TO 10
      GO TO 60
      END
```

```
      SUBROUTINE REG2 (ITR)
C*       SIMULATES OPERATION OF REGION2                                        *
      COMMON /CHANL1/ BUSY1,PROCH1,ESERV1,NSERV1
      COMMON /CHANL2/ BUSY2,PROCH2,ESERV2,NSERV2
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /RAND/ IX,IY
      COMMON /XACT/ IXACT
      COMMON /REGN2/ BREG2,INPRO2,EPROC2,PTIME2,DBC2,CHAN2,USE2,NXTMS2,P
     1PROC2,TOGO2,NPROG2,NDBC2
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      COMMON /RUSE/ PTYPE1,PTYPE2,TIME1,TIME2
      COMMON /MAIN1/ EOSCH1,ENSCH1,BUILT1,MSGSC1,IRCODE,BUILD1
      COMMON /MAIN2/ EOSCH2,ENSCH2,BUILT2,MSGSC2,BUILD2
      COMMON /DBASE/ DBTAB
      INTEGER DBC2,NXTMS2,PTYPE1,PTYPE2
      INTEGER BREG2,CHAN2,USE2,TOGO2,DBTAB(41,2)
      INTEGER BUSY1,BUSY2,PROCH1,PROCH2,ESERV1,ESERV2
      INTEGER EPROC2,PTIME2,PPROC2,TIME1,TIME2
      INTEGER TABLE
      INTEGER EOSCH1,EOSCH2,ENSCH1,ENSCH2,BUILT1,BUILT2,BUILD1,BUILD2
      DIMENSION TABLE(200,30), DBC2(48), IXACT(27,75), CVALS(30,10)
      IRCODE=2
      GO TO (10,30,40,60,80,120), ITR
10    CALL NXTMSG (2)
      IF (MSGSC2.EQ.0) RETURN
      JA=LOCATE(MSGSC2)
      TABLE(JA,6)=2
      NPROG2=TABLE(JA,16)
      TOGO2=TABLE(JA,13)
      NXTMS2=MSGSC2
      IF (TABLE(JA,10).EQ.0) GO TO 20
      ENSCH2=CLOCK
```

```
      TABLE(JA,7)=4
      BREG2=1
      RETURN
20    CALL CHNL1 (MSGSC2)
      TABLE(JA,7)=3
      BREG2=1
      RETURN
30    EOSCH2=CLOCK+10
      BREG2=1
      JA=LOCATE(MSGSC2)
      TABLE(JA,7)=4
      RETURN
40    JA=LOCATE(NXTMS2)
      TABLE(JA,7)=5
      BREG2=1
      IF (TABLE(JA,10).EQ.0) GO TO 50
      BUILT2=CLOCK
      RETURN
50    CONTINUE
      BUILD2=10
      BUILT2=BUILD2+CLOCK
      RETURN
60    BREG2=1
      INPRO2=NXTMS2
      JA=LOCATE(INPRO2)
      TABLE(JA,7)=6
      NDBC2=TABLE(JA,5)
      INDX=TABLE(JA,2)
      USE2=TABLE(JA,9)
      PTYPE2=TABLE(JA,16)
      PTIME2=TABLE(JA,4)
      DO 70 ID=1,47
```

```
      DBC2(ID)=IXACT(INDX,ID+15)
70    CONTINUE
80    CONTINUE
      IF (NDBC2.LE.0) GO TO 110
      IDB=DBC2(NDBC2)
      CHAN2=DBTAB(IDB,2)
C*****************************************************************
C                    GENERATE PARTIAL PROCESSING TIME            *
C*****************************************************************
      CALL RNDNO (IX,IY,PT)
      IX=IY
      JA=LOCATE(INPRO2)
      PPROC2=PT*PTIME2
      IND=TABLE(JA,2)
      CVALS(IND,7)=CVALS(IND,7)+PPROC2
      PTIME2=PTIME2-PPROC2
      PPROC2=CLOCK+PPROC2
      IF (PTIME2.LE.0.0) GO TO 90
      IF (NDBC2.LE.0) GO TO 110
      RETURN
90    CONTINUE
      IF (NDBC2.LE.0) GO TO 110
      DO 100 JQ=1,NDBC2
      IF (CHAN2.EQ.1) CALL CHNL1 (INPRO2)
      IF (CHAN2.EQ.2) CALL CHNL2 (INPRO2)
100   CONTINUE
      NDBC2=0
      RETURN
110   EPROC2=CLOCK+PTIME2
      PPROC2=ENDSIM*10.
      TABLE(JA,5)=0
      RETURN
```

```
120   BREG2=0
      CLOCK=EPROC2
      EPROC2=ENDSIM*10.
      JA=LOCATE(INPRO2)
      TINSYS=CLOCK-TABLE(JA,3)
      IND=TABLE(JA,2)
      CVALS(IND,8)=CVALS(IND,8)+TINSYS
      TABLE(JA,20)=0
      INPRO2=0
      NPROG2=0
      IF (TOGO2.LE.0.OR.TABLE(JA,13).LE.1) GO TO 10
      CALL REUSE (2)
      TOGO2=TOGO2-1
      IF (NXTMS2.EQ.0) GO TO 10
      GO TO 60
      END
```

```
      SUBROUTINE REUSE (IREG)
C        SELECTS NEXT MESSAGE IF PROGRAM IS REUSABLE
      COMMON /$STAT/ CVALS,NOREG1,NOREG2,START,MSGT
      COMMON /REGN1/ BREG1,INPRO1,EPROC1,PTIME1,DBC1,CHAN1,USE1,NXTMS1,P
     1PROC1,TOGO1,NPROG1,NDBC1
      COMMON /REGN2/ BREG2,INPRO2,EPROC2,PTIME2,DBC2,CHAN2,USE2,NXTMS2,P
     1PROC2,TOGO2,NPROG2,NDBC2
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      COMMON /MQUE/ MSGQ,MSGQLN
      COMMON /RUSE/ PTYPE1,PTYPE2,TIME1,TIME2
      DIMENSION MSGQ(200), TABLE(200,30), CVALS(30,10), MSGT(27,3)
      INTEGER PTYPE1,PTYPE2,DBC1(48),DBC2(48)
      NOREG1=NOREG1+1
      GO TO (10,60), IREG
10    IF (MSGQLN.EQ.0) GO TO 90
      DO 20 I=1,MSGQLN
      JA=LOCATE(MSGQ(I))
      IF (TABLE(JA,16).EQ.PTYPE1) GO TO 30
20    CONTINUE
      NXTMS1=0
      RETURN
30    NXTMS1=TABLE(JA,1)
      TABLE(JA,6)=1
      NOREG1=NOREG1+1
40    DO 50 K=I,199
      MSGQ(K)=MSGQ(K+1)
50    CONTINUE
      MSGQLN=MSGQLN-1
      TIMSGQ=CLOCK-TABLE(JA,11)
      IND=TABLE(JA,2)
      CVALS(IND,6)=CVALS(IND,6)+TIMSGQ
      TABLE(JA,11)=CLOCK
```

```
      MSGT(TABLE(JA,2),IREG)=MSGT(TABLE(JA,2),IREG)+1
      RETURN
60    IF (MSGQLN.EQ.0) GO TO 100
      DO 70 I=1,MSGQLN
      JA=LOCATE(MSGQ(I))
      IF (TABLE(JA,16).EQ.PTYPE2) GO TO 80
70    CONTINUE
      NXTMS2=0
      RETURN
80    NXTMS2=TABLE(JA,1)
      TABLE(JA,6)=2
      NOREG2=NOREG2+1
      GO TO 40
90    NXTMS1=0
      RETURN
100   NXTMS2=0
      RETURN
      END
```

```
      SUBROUTINE ENVIR1
C*       GENERATES EXTERNAL TRANSACTIONS ON CHANNEL1                       *
      COMMON /ENV/ TNVIR1,TNVIR2,NVIR1,NVIR2
      COMMON /RAND/ IX,IY
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      DIMENSION TABLE(200,30)
      INTEGER TNVIR1,TNVIR2,TABLE
      NUM=NUM+1
      DO 10 I=1,200
      INSYS=TABLE(I,20)
      IF (INSYS.EQ.0) GO TO 20
10    CONTINUE
      RETURN
20    TABLE(I,1)=NUM
      TABLE(I,2)=28
      TABLE(I,6)=0
      TABLE(I,7)=7
      TABLE(I,12)=1
      TABLE(I,19)=50
      TABLE(I,20)=1
      TABLE(I,5)=1
      CALL CHNL1 (NUM)
      CALL RNDNO (IX,IY,T)
      IX=IY
      NVIR1=NVIR1+1
      TNVIR1=(-1.0)*((1/21390.)*3.6E06*ALOG(T))+CLOCK
      RETURN
      END
```

```
      SUBROUTINE ENVIR2
C*       GENERATES EXTERNAL TRANSACTIONS ON CHANNEL2                         *
      COMMON /ENV/ TNVIR1,TNVIR2,NVIR1,NVIR2
      COMMON /RAND/ IX,IY
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      INTEGER TNVIR1,TNVIR2,TABLE
      DIMENSION TABLE(200,30)
      NUM=NUM+1
      DO 10 I=1,200
      INSYS=TABLE(I,20)
      IF (INSYS.EQ.0) GO TO 20
10    CONTINUE
      RETURN
20    TABLE(I,1)=NUM
      TABLE(I,2)=29
      TABLE(I,6)=0
      TABLE(I,7)=7
      TABLE(I,12)=1
      TABLE(I,19)=40
      TABLE(I,20)=1
      TABLE(I,5)=1
      CALL CHNL2 (NUM)
      CALL RNDNO (IX,IY,T)
      IX=IY
      NVIR2=NVIR2+1
      TNVIR2=(-1.0)*((1/21390.)*3.6E06*ALOG(T))+CLOCK
      RETURN
      END
```

```
      FUNCTION LOCATE (IDEN)
C*       LOCATES ENTRIES IN MASTER TABLE                                 *
      COMMON /ALL/ CLOCK,NUM,TABLE,ENDSIM
      DIMENSION TABLE(200,30)
      INTEGER TABLE
      DO 10 I=1,200
      NUMB=TABLE(I,1)
      INSYS=TABLE(I,20)
      IF (NUMB.EQ.IDEN.AND.INSYS.GE.1) GO TO 20
10    CONTINUE
      LOCATE=0
      CALL STATS
      RETURN
20    LOCATE=I
      RETURN
      END
```

```
      SUBROUTINE RNDNO (IX,IY,YFL)
C*       RANDOM NUMBER GENERATOR                                       *
      IY=IX*65539
      IF (IY) 10,20,20
10    IY=IY+2147483647+1
20    YFL=IY
      YFL=YFL*.4656613E-9
      RETURN
      END
```

```
      SUBROUTINE SHELL (MAT,ICOL,N,NCOLS)
C***   PERFORMS SHELL SORT                                          *
C
C         MAT = MATRIX TO BE SORTED IN ASCENDING ORDER.
C         ICOL = COLUMN TO BE SORTED ON.
C         N = NUMBER OF ROWS IN MAT.
C         NCOLS = NUMBER OF COLUMNS IN MAT.
C*******************************************************************
      DIMENSION MAT(200,5), ITEMP(100)
      M=N
10    M=M/2
      IF (M.EQ.0) GO TO 60
      K=N-M
      J=1
20    I=J
30    L=I+M
      IF (MAT(I,ICOL).GE.MAT(L,ICOL)) GO TO 50
      DO 40 KX=1,NCOLS
      ITEMP(KX)=MAT(I,KX)
      MAT(I,KX)=MAT(L,KX)
      MAT(L,KX)=ITEMP(KX)
40    CONTINUE
      I=I-M
      IF (I.GE.1) GO TO 30
50    J=J+1
      IF (J.GT.K) GO TO 10
      GO TO 20
60    RETURN
      END
```

APPENDIX C

DATA FOR REGRESSION MODELS

Table C.1 Input Data Used for Regression Model. Existing Configuration

X_1	X_2	Y
1.00	1.00	4.25
1.10	1.00	5.44
1.25	1.00	7.19
1.50	1.00	10.42
1.00	1.10	4.70
1.80	1.25	8.10
1.00	1.50	10.60
1.10	1.10	5.26
1.25	1.25	7.52
1.50	1.50	10.83
1.30	1.20	8.15
1.40	1.15	6.46
1.15	1.15	6.10
1.20	1.15	8.19

EXISTING CONFIGURATION

GENERAL LINEAR MODELS PROCEDURE

DEPENDENT VARIABLE: Y

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PR > F	R-SQUARE	C.V.
MODEL	3	51.98322577	17.32774192	19.10	0.0002	0.851391	12.9210
ERROR	10	9.07360994	0.90736099		STD DEV		Y MEAN
CORRECTED TOTAL	13	61.05683571			0.95255498		7.37214286

SOURCE	DF	TYPE I SS	F VALUE	PR > F	DF	TYPE IV SS	F VALUE	PR > F
X1	1	17.22493747	18.98	0.0014	1	15.88799628	17.51	0.0019
X2*X2	1	23.28521150	25.66	0.0005	1	17.91954512	19.75	0.0012
X3	1	11.47307680	12.64	0.0052	1	11.47307680	12.64	0.0052

PARAMETER	ESTIMATE	T FOR H0: PARAMETER=0	PR > |T|	STD ERROR OF ESTIMATE
INTERCEPT	-21.55869361	-4.07	0.0023	5.30161923
X1	35.00570597	4.18	0.0019	8.36554672
X2*X2	15.16635861	4.44	0.0012	3.41277678
X3	-24.30913271	-3.56	0.0052	6.83627567

OBSERVATION	OBSERVED VALUE	PREDICTED VALUE	RESIDUAL
1	4.25000000	4.30423627	-0.05423627
2	5.44000000	5.37389559	0.06610441
3	7.19000000	6.97838158	0.21161842
4	10.42000000	9.65252490	0.76747510
5	4.71000000	5.06526031	-0.35526031
6	8.10000000	6.75803181	1.34196819
7	10.60000000	11.10762016	-0.50762016
8	5.26000000	5.88482650	-0.62482650
9	7.52000000	7.91285433	-0.39285433
10	10.83000000	10.37862363	0.45137637
11	8.15000000	7.86603353	0.28396647
12	6.46000000	8.36910036	-1.90910036
13	6.10000000	6.60654952	-0.50654952
14	8.19000000	6.95905969	1.23094031

SUM OF RESIDUALS	-0.00000000
SUM OF SQUARED RESIDUALS	9.07360994
SUM OF SQUARED RESIDUALS - ERROR SS	0.00000000
FIRST ORDER AUTOCORRELATION	-0.11464054
DURBIN-WATSON D	2.04191379

Table C.2 Input Data Used for Regression Model. Third Region

X_1	X_2	Y
1.00	1.00	3.57
1.10	1.00	4.70
1.25	1.00	6.90
1.50	1.00	8.30
1.00	1.10	7.30
1.00	1.25	7.90
1.00	1.50	9.40
1.10	1.10	7.30
1.25	1.25	9.80
1.50	1.50	10.10
1.30	1.20	10.38
1.40	1.15	7.89
1.15	1.15	8.75
1.20	1.15	9.86

THIRD REGION

GENERAL LINEAR MODELS PROCEDURE

DEPENDENT VARIABLE: Y

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PR > F	R-SQUARE	C.V.
MODEL	5	44.78675886	8.95735177	9.78	0.0029	0.859435	11.9451
ERROR	8	7.32513400	0.91564175		STD DEV		Y MEAN
CORRECTED TOTAL	13	52.11189286			0.95689171		8.01071429

SOURCE	DF	TYPE I SS	F VALUE	PR > F	DF	TYPE IV SS	F VALUE	PR > F
X1*X1	1	10.07532330	11.00	0.0106	1	0.60101605	0.66	0.4413
X2*X2	1	18.05385739	19.72	0.0022	1	3.24409953	3.54	0.0966
X1	1	7.65434579	8.36	0.0202	1	1.41338521	1.54	0.2493
X2	1	6.23812334	6.81	0.0311	1	5.77627974	6.31	0.0363
X3*X3	1	2.76510904	3.02	0.1205	1	2.76510904	3.02	0.1205

PARAMETER	ESTIMATE	T FOR H0: PARAMETER=0	PR > |T|	STD ERROR OF ESTIMATE
INTERCEPT	-67.64153584	-3.84	0.0050	17.68333253
X1*X1	-9.58073217	-0.81	0.4413	11.82547134
X2*X2	-21.46196426	-1.88	0.0966	11.40210866
X1	35.67176824	1.24	0.2493	28.87256046
X2	69.20550935	2.51	0.0363	27.55366374
X3*X3	-1.98465712	-1.74	0.1205	1.14206849

OBSERVATION	OBSERVED VALUE	PREDICTED VALUE	RESIDUAL
1	3.57000000	4.20638817	-0.63638817
2	4.70000000	5.36663324	-0.66663324
3	6.90000000	6.67079874	0.22920126
4	8.30000000	7.68755566	0.61244434
5	7.30000000	6.20514861	1.09485139
6	7.90000000	8.32104097	-0.42104097
7	9.40000000	9.50286610	-0.10286610
8	7.30000000	7.27607030	0.02392970
9	9.80000000	10.15549363	-0.35549363
10	10.10000000	9.88098684	0.21901316
11	10.35000000	9.91184657	0.43815343
12	7.89000000	9.65916289	-1.76916289
13	8.75000000	8.47218961	0.27781039
14	9.86000000	8.83163868	1.02836132

SUM OF RESIDUALS	-0.00000000
SUM OF SQUARED RESIDUALS	7.32513400
SUM OF SQUARED RESIDUALS - ERROR SS	-0.00000000
FIRST ORDER AUTOCORRELATION	-0.05379359
DURBIN-WATSON D	1.89670502

Table C.3 Input Data Used for Regression Model.
Third Channel

X_1	X_2	Y
1.00	1.00	1.68
1.10	1.00	1.70
1.25	1.00	1.82
1.50	1.00	5.64
1.00	1.10	1.71
1.00	1.25	1.75
1.00	1.50	1.79
1.10	1.10	1.73
1.25	1.25	1.88
1.50	1.50	7.75
1.30	1.20	1.99
1.40	1.15	2.33
1.15	1.15	1.74
1.20	1.15	1.81
1.00	2.00	8.11

THIRD CHANNEL

GENERAL LINEAR MODELS PROCEDURE

DEPENDENT VARIABLE: Y

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PR > F	R-SQUARE	C.V.
MODEL	5	69.82251749	13.96450350	50.02	0.0001	0.965261	18.2500
ERROR	9	2.51285584	0.27920620		STD DEV		Y MEAN
CORRECTED TOTAL	14	72.33537333			0.52839966		2.89533333

SOURCE	DF	TYPE I SS	F VALUE	PR > F	DF	TYPE IV SS	F VALUE	PR > F
X1*X1	1	12.22394132	43.76	0.0001	1	4.73871603	16.97	0.0026
X2*X2	1	43.92792779	157.33	0.0001	1	9.89982346	35.46	0.0002
X1	1	6.59738506	23.63	0.0009	1	4.86893979	17.44	0.0024
X2	1	5.16506758	18.50	0.0020	1	7.04189510	25.22	0.0007
X3*X3	1	1.90819573	6.83	0.0281	1	1.90819573	6.83	0.0281

PARAMETER	ESTIMATE	T FOR H0: PARAMETER=0	PR > |T|	STD ERROR OF ESTIMATE
INTERCEPT	53.99045151	5.89	0.0002	9.17317430
X1*X1	24.06738569	4.12	0.0026	5.84199402
X2*X2	10.22281619	5.95	0.0002	1.71676505
X1	-58.18840159	-4.18	0.0024	13.93419096
X2	-29.49119054	-5.02	0.0007	5.87431592
X3*X3	1.57612890	2.61	0.0281	0.60366159

OBSERVATION	OBSERVED VALUE	PREDICTED VALUE	RESIDUAL
1	1.68000000	2.16899015	-0.48899015
2	1.70000000	1.73570605	-0.03570605
3	1.82000000	2.04749171	-0.22749171
4	5.64000000	5.13168258	0.50831742
5	1.71000000	1.69702757	0.01297243
6	1.75000000	1.43161163	0.31838837
7	1.79000000	2.16932624	-0.37932624
8	1.73000000	1.33334095	0.39665905
9	1.88000000	1.80944303	0.07055697
10	7.75000000	7.59784508	0.15215492
11	1.99000000	2.17908440	-0.18908440
12	2.33000000	3.38247393	-1.05247393
13	1.74000000	1.25610534	0.48389466
14	1.81000000	1.41983445	0.39016555
15	6.11000000	6.07003488	0.03996512

SUM OF RESIDUALS	0.00000000
SUM OF SQUARED RESIDUALS	2.51285584
SUM OF SQUARED RESIDUALS - ERROR SS	-0.00000000
FIRST ORDER AUTOCORRELATION	-0.18665560
DURBIN-WATSON D	2.25956269

Table C.4 Input Data Used for Regression Model.
Three Regions - Three Channels

X_1	X_2	Y
1.00	1.00	1.93
1.10	1.00	2.01
1.25	1.00	2.22
1.50	1.00	5.68
1.00	1.10	1.96
1.00	1.25	1.99
1.00	1.50	2.11
1.10	1.10	2.07
1.25	1.25	2.47
1.50	1.50	6.45
1.30	1.20	2.68
1.40	1.15	3.48
1.15	1.15	2.16
1.20	1.15	2.24

THREE CHANNELS - THREE REGIONS

GENERAL LINEAR MODELS PROCEDURE

DEPENDENT VARIABLE: Y

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE	PR > F	R-SQUARE	C.V.
MODEL	5	26.46737273	5.29347455	78.12	0.0001	0.979931	9.2376
ERROR	8	0.54206298	0.06775787		STD DEV		Y MEAN
CORRECTED TOTAL	13	27.00943571			0.26030342		2.81785714

SOURCE	DF	TYPE I SS	F VALUE	PR > F	DF	TYPE IV SS	F VALUE	PR > F
X1*X1	1	20.98389956	309.69	0.0001	1	2.97416768	43.89	0.0002
X2*X2	1	1.23251123	18.19	0.0027	1	0.03813907	0.56	0.4746
X1	1	3.94069522	58.16	0.0001	1	2.54399736	37.55	0.0003
X2	1	0.07612414	1.12	0.3201	1	0.06207728	0.92	0.3665
X3*X3	1	0.23414258	3.46	0.1001	1	0.23414258	3.46	0.1001

PARAMETER	ESTIMATE	T FOR H0: PARAMETER=0	PR > |T|	STD ERROR OF ESTIMATE
INTERCEPT	33.21580316	6.90	0.0001	4.81040013
X1*X1	21.31269604	6.63	0.0002	3.21688510
X2*X2	2.32705912	0.75	0.4746	3.10171767
X1	-48.12611909	-6.13	0.0003	7.85420796
X2	-7.17435544	-0.96	0.3665	7.49542811
X3*X3	0.57752336	1.66	0.1001	0.31067710

OBSERVATION	OBSERVED VALUE	PREDICTED VALUE	RESIDUAL
1	1.93000000	2.13260715	-0.20260715
2	2.01000000	1.91694132	0.09305868
3	2.22000000	2.41432579	-0.19432579
4	5.68000000	5.43232185	0.24767815
5	1.96000000	2.02513393	-0.06513393
6	1.99000000	1.97284593	0.01715407
7	2.11000000	2.17615753	-0.06615753
8	2.07000000	1.83493687	0.23506313
9	2.47000000	2.43729657	0.03270343
10	6.45000000	6.37825247	0.07174753
11	2.68000000	2.81750410	-0.13750410
12	3.48000000	3.93614590	-0.45614590
13	2.16000000	1.89392561	0.26607439
14	2.24000000	2.08160497	0.15839503

SUM OF RESIDUALS	0.00000000
SUM OF SQUARED RESIDUALS	0.54206298
SUM OF SQUARED RESIDUALS - ERROR SS	-0.00000000
FIRST ORDER AUTOCORRELATION	-0.26592539
DURBIN-WATSON D	2.37733025

The vita has been removed from
the scanned document

An Analysis of IMS Through Simulation

by

James W. Chrissis

ABSTRACT

This thesis presents the development and validation of a simulation model for an on-line IMS/VS configuration. A conceptual queueing model is developed from an actual system configuration and the simulation model parallels this conceptualization. The validation of the simulation model is carried out in three phases utilizing operational data from IMS logs and system monitors. This model is then used to evaluate system performance under a variety of operating configurations and system loads. A configuration which results in improved IMS response is the objective of the simulation analysis.