PERFORMANCE MODELING AND EVALUATION OF SAMIS COMPUTER NETWORK USING PAWS

Mehmet Ufuk Çağlayan*

Department of Computer Engineering
Bogazici University
Bebek, Istanbul, Turkey

الخلاصـــة :

أهم أهداف هذا البحث هو إحداث برنامج مساعد يُسمى PAWS للنمذجة وتقييم أداء شبكة نظام الكمبيوتر الخاص بوزارة الداخلية في المملكة العربية السعودية .

وحتى يتّـم إيضاح طريقة تطبيق هذا البرنامج لمنذجة مكونات الكمبيوتر الفيزيائية والبرامج المساعدة ، فلقد تـمّـت التجربة على شبكة الكمبيوتر في وزارة الداخلية من أجل البحث والتقييم . والدراسة نشرح باختصار بنيات لغة المحاكاة PAWS وسهولة استخدامها ووظائفها ، وميزات استخدام النظام .

وأخيراً يتضمن البحث النتائج التي حصلنا عليها من عملية المحاكاة والاقتراحات التي من شأنها تحسين أداء النظام .

ABSTRACT

A high level performance modeling and evaluation software tool called PAWS is introduced, and its application to modeling the hardware and software of the SAMIS Online System to evaluate its performance is demonstrated. Properties of the SAMIS Online System are first presented. Some of the important simulation language structures available in PAWS are briefly described, and their ease of use and functionality are discussed. Problems encountered during the performance modeling process are discussed and procedures that are adopted to create a successful model representing the actual system are presented. Two models using the PAWS language are also described together with the modeling assumptions and the reasoning behind the assumptions. Finally, the results obtained through modeling and suggestions to improve the system performance are presented.

*Present address:
ZXT-Informatik-Technologie
BASF AG
D-6700 Ludwigshafen
West Germany

PERFORMANCE MODELING AND EVALUATION OF SAMIS COMPUTER NETWORK USING PAWS

1. INTRODUCTION

The goal of this paper is to introduce a high level performance modeling and evaluation software tool called PAWS, and to demonstrate its application to modeling the hardware and software of the SAMIS (Saudi Arabian Ministry of Interior System) Online System for the purpose of evaluating its performance. The paper summarizes the work done in one independent task of a research project carried out to improve the operational effectiveness of the SAMIS computer network.

The research project has been completed in eighteen months and has fully been documented in three progress reports [1–3], a final report [4], and in [5]. It consisted of two major and independent tasks, namely the development of Arabic input/output devices, and performance engineering and system modeling. Only the work done in the latter task is presented in this paper. The objective of the latter task was to develop and validate a simulation model to predict system performance, identify potential bottlenecks, explore alternative configurations, and help fine-tune the system for best performance.

The SAMIS computer network is a large hardware/software system that is operated by the National Information Center, Ministry of Interior, Riyadh, Saudi Arabia. It is used to acquire, maintain, and provide vital information about the citizens and the expatriates in the country. A part of this network, called the SAMIS Online System, has been modeled and its performance has been evaluated to develop an understanding of the interaction of all hardware and software components, to verify whether it can provide reasonable service to its users, and to plan for the furture expansion of the network. The general structure and properties of the SAMIS Online System will be given in Section 2.

PAWS (Performance Analyst's Workbench System) is a software package incorporating a very high level simulation language and associated tools [6, 7]. PAWS enables a modeler to concentrate on the high level details of modeling rather than the low level details of coding. A model is first pictorially constructed in a notation called Information Processing Graphs, then a simulation program is coded in PAWS simulation language. Some important primitive structures available in PAWS will be briefly described, and their ease of use and functionality will be discussed in Section 3. Also, a number of concepts available in PAWS and the statistical output generated by the PAWS package will be evaluated.

In Section 4, the modeling approach, the problems encountered during the performance modeling process, and procedures that are adopted to create a successful model representing the actual system will be presented. Modeling assumptions and the reasoning for the assumptions will also be provided. Two models using the PAWS language will be described in Section 5. Finally, the results that are obtained through modeling process and suggestions made to improve the network performance will be presented in Section 6.

2. SAMIS ONLINE SYSTEM

The SAMIS computer network is a large information system operated by the National Information Center, Ministry of Interior, Riyadh, Saudi Arabia. The Online System and the Administrative System are the two major components of the SAMIS computer network. Since they are mostly independent, separate UNIVAC mainframe computer systems are allocated to each, and there exists a third backup mainframe which can take over either system in case of failures. It has been decided to consider only the Online System for modeling, because of the online nature of applications and performance problems already encountered in terms of poor response time.

The SAMIS Online System hardware consists of dual CPU Univac 1100/82 computer system with 1.5M words (36 bits per word) main memory, 64 disk drives (78K pages each, 512 words per page) in two large disk subsystems, 9 fast drums (262K words each), and 15 slow drums (2M words each) in three drum subsystems, around 1000 dual language terminals, and other associated components such as tape drives, printers, communication controllers, *etc.* [4].

The SAMIS Online System uses the CSTS-II operating system developed by Computer Science Corporation. The Interaction Management Protocol (IMP) subsystem of CSTS-II allows transaction processing by switching a large number of terminals among an integrated collection of application tasks. A relational database system called MANAGE provides file and database operations to support both the operating system and the application system.

The application system of the Online System consists of eleven subsystems, namely Miscreant, Passport Issuance, Criminal Records, Alien Control, Border Control, Drivers License, Motor Vehicle, Pilgrim Control, Civil Registry, Micrographics, and Message Communication subsystems. Each subsystem supports a variety of functions that consist of a specific transaction flow. Around 120 different functions are supported in 11 subsystems. The SAMIS computer network terminals are configured to enable the selection of a specific set of functions only. The invocation of a function from a terminal starts the execution of a sequence of application system tasks, where the execution sequence is well defined in the thread chart description of the SAMIS Online System [4, 5].

3. OVERVIEW OF PAWS

Performance Analyst's Workbench System (PAWS)* is a software package providing state-of-the-art simulation of computer and communication systems. The PAWS language uses high level concepts and primitives, though it is simple, powerful, and easy to learn [6, 7]. It enables the modeler to concentrate on system structure and dynamics at a high level, without getting involved in model implementation intricacies.

The PAWS package was selected for a variety of reasons. First, it was one of the best packages available at the time and was used by many other teams in complicated simulation studies. Second, the package provided a very high level modeling language, therefore a reduced modeling time was expected. Third, the package came in source code, therefore enabling the modeler to make changes in the source code. And last, modelers were already familiar with the package and re-training to use the package properly was minimal.

In order to construct a PAWS model, the user must first abstract, from the real system to be modeled, a pictorial representation of the system called an Information Processing Graph (IPG). The user next translates the IPG into a complete PAWS model description, which PAWS simulates to obtain the requested performance statistic estimates.

IPGs are based on the concept of transactions which are processed at some nodes. The category of a transaction represents its type. It is static during simulation. The phase of a transaction is used to represent dynamic aspects of it. Transactions move from a node to another according to the node topology of a PAWS model.

Different types of nodes exist for different types of transaction processing, such as resource management nodes, routing nodes, arithmetic nodes, interrupt nodes, and user nodes. Resource management nodes are classified to represent active resources and passive resources. An active resource is represented by a SERVICE node, which is basically a server with a queue. Passive resources are either memories or tokens, and represent objects that must be possessed by a transaction to do work. Passive resources are acquired and later released, therefore they are represented by a pair of nodes. For example, tokens are acquired at ALLOCATE nodes and released at RELEASE nodes, and memories at GETMEN nodes and RELMEM nodes respectively.

Routing nodes are used to create, destroy, or alter the paths of transactions. Transactions of any category/phase can be created at a SOURCE node according to a specified interarrival time distribution. They are destroyed at SINK nodes. FORK, JOIN, SPLIT, and BRANCH are routing nodes to move transactions according to their categories and/or phases.

Arithmetic nodes are used to carry out computations on PAWS variables and consist of COMPUTE and CHANGE nodes. A COMPUTE node is similar to an ordinary FORTRAN assignment statement. A CHANGE node is used to change the phase of a transaction in a probabilistic manner.

An interrupt node is used by one transaction to interrupt the processing of another transaction. The interrupted transaction immediately departs the node with a new phase assigned to it by its interrupter. User nodes allow the modeler to interface FORTRAN subroutines with a PAWS model. This facility makes PAWS an extensible system.

A PAWS program mainly consists of: (*a*) a DECLARE section, in which all variables, nodes, categories, tokens, and memories are given;

*IRA, the developer of PAWS, changed its name to SES, Inc. and PAWS has developed into DEW (Design Evaluation Workbench). DEW is based on C, rather than Fortran as with PAWS. DEWS runs much faster than PAWS and provides more functions and a better graphical interface.

(*b*) a TOPOLOGY section in which edges connecting the nodes are defined; (*c*) a DEFINE section in which each node is defined in detail; (*d*) an INITIAL section in which the initial conditions are declared; (*e*) a STATISTICS section in which the type and parameters of statistics to be collected are defined; and (*f*) a RUN section that actually causes PAWS to simulate a model.

The PAWS language is a declarative one, and therefore it is easy to learn and use. The interested reader can consult [6, 7] for more details since it is impossible to include all aspects of PAWS in this paper.

The PAWS software package has been run on a DEC VAX-11/780 VMS computer system, and it is available for several other computer systems.

4. MODELING APPROACH AND ASSUMPTIONS

The use of modeling and simulation *versus* measurement as different methods of performance evaluation is well known [8]. Modeling and simulation was chosen since the system whose performance to be evaluated is very complex. Also the system was partially operational, therefore the results of any measurement study would not be indicative.

Initially, a number of trips were made to the National Information Center (NIC) to obtain system documentation on SAMIS hardware and software configuration, and to meet and interview NIC key personnel for the purpose of getting answers to specific questions regarding system details and recent system modifications.

Most of the existing SAMIS documentation related to the project [1–4] were studied to identify the hardware and software components, and their functional relationships. The initial effort was concentrated on understanding the details of the CSTS-II operating system, the MANAGE database management system, and the components of the application software system.

Since some subsystems were not operational at the time of modeling and not all subsystems were equally important because of their database size and/or frequency of usage, it was decided to restrict the initial modeling effort to cover the three most important subsystems, namely Civil Registry, Alien Control, and Border Control.

Modeling of the communication hardware and software components was not considered due to the facts that not all components were operational, and few problems were expected to be found there. The model was constructed to take into account the communication system in terms of communication delays and the operator "think time" involved, both of which are parameters affecting the transaction arrival and departure rates. The sensitivity of the model to these two parameters need to be evaluated to find out whether it is worth the effort to model the communication system separately.

The SAMIS Online System transaction volume, broken down with respect to the Online System subsystems, was computed and projected for the year 1990 [1]. It was calculated that the transaction rate will be around 6 transactions per second and 85% of the transaction load will be generated by four subsystems, out of eleven subsystems.

Important observations that were made during interviews and document evaluation were as follows.

1. The mainframe on which the Online System is operating seemed to have a relatively small memory size compared with the size and complexity of the operating system, the database management system, and the application subsystems.
2. Similarly, the amount of auxiliary storage provided in terms of fast drums seemed to be small. Also, the specific way these drums were used by the system, namely the "file promotion" operation, seemed to have a significant negative effect on the system performance when very large files are considered.

These observations implied that the amount of main memory and the drum bandwidth may be performance bottlenecks in the Online System, especially after all the subsystems are operational with fully developed databases. Therefore, it was decided that the actual effects of these two important system properties on system performance in terms of response time and system throughout needed to be quantified through the modeling process.

After a sufficient understanding of the SAMIS Online System had been developed, a two-level approach was agreed upon to construct a performance model. This approach has resulted in two simulation models for performance evaluation, namely the disk–drum model and the data-driven model. These models were aimed at representing the

SAMIS Online System's performance at different levels of detail. The next section will give a description of both of these models.

5. SIMULATION MODELS IN PAWS

Initially, a disk–drum model of the SAMIS Online System was constructed. Although it was expected that this model would not precisely reflect the intricacies of the application system, it was nevertheless constructed to have an overall understanding of the underlying computer hardware, and its interaction with the operating system and the application system in a global manner. The disk–drum model incorporated the following properties of the Online System:

All hardware components such as CPUs, memory, channels, drums, disks.

Physical properties and parameters of hardware components in terms of quantity, size, speed, *etc.*

Configuration of the hardware components such as which channels control which drums and/or disks.

Properties and parameters of the operating system components, *i.e.* scheduling, memory management, queuing disciplines, and input/output device allocation strategies.

Parameters representing the overall average behavior of all application software system components, such as input transaction arrival time distribution, memory request patterns, frequency of disk or drum input/output per transaction, *etc.*

The information processing graph of the disk–drum model is shown in Figure 1. Modeling assumptions and scenarios are as follows:

Input transaction load is assumed to consist of PAWS transactions in 10 different categories, namely ALIEN, BORDER, CIVIL, CRIM, DRIVE, MISCR, MOTOR, PASS, PLGRM, and OTHER. Each category represents an Online System subsystem.

Transaction interarrival time is assumed to be exponentially distributed. The mean interarrival time in milliseconds is set according to the overall transaction rate, in number of transactions per second, ranging from 2 to 10 trans s^{-1}. Distribution of categories is assumed to be uniform, representing the fact that there is equal load on every subsystem. The load on each subsystem should actually be characterized by measuring the frequency of transactions during a representative day.

All transaction interdependencies are ignored, since it is not possible to represent such dependencies in the disk–drum model. The transaction dependency means that a transaction of a certain subsystem is not only created by the operator of that subsystem directly, but also by other subsystems.

2282 pages representing the total user area of main memory is allocated by best fit. Main memory requests are served First-Come-First-Serve (FCFS).

The memory requests are assumed to be uniformly distributed between 6 and 108 pages of code that includes both instruction and data. It is assumed that transactions that arrive following a transaction that is already in the system and of the same category request only one page of data area since tasks representing the subsystems are reentrant. The assumption on uniform distribution of memory requests, irrespective of transaction categories, is not a realistic one, since different category transactions request different amounts of memory. The model can easily be updated once request distributions per category are known.

Dual CPU, single scheduling algorithm results in PAWS node with a single input queue and two servers. Queuing discipline is FCFS.

Scheduling algorithm is simplified to FCFS. CPU times are assumed to be exponentially distributed with the same mean, irrespective of transaction categories. Mean values ranging from 1 to 10 milliseconds have been studied in the model.

Average number of disk I/O operations and average number of drum I/O operations per transaction, irrespective of the transaction type, were computed by taking a weighted average based on file sizes expected in 1990 and file types as specified in the design documents. Whether the files are of type index or hashed was taken into account since this fact very much affected the number of disk operations. It was found that a transaction issues, on the average, 28 drum and 9 disk I/O operations every time it enters the system.

Based on the average number of disk–drum I/O operations per transaction, the probability that a transaction will request disk or drum I/O operation, after its current CPU time is over is computed in

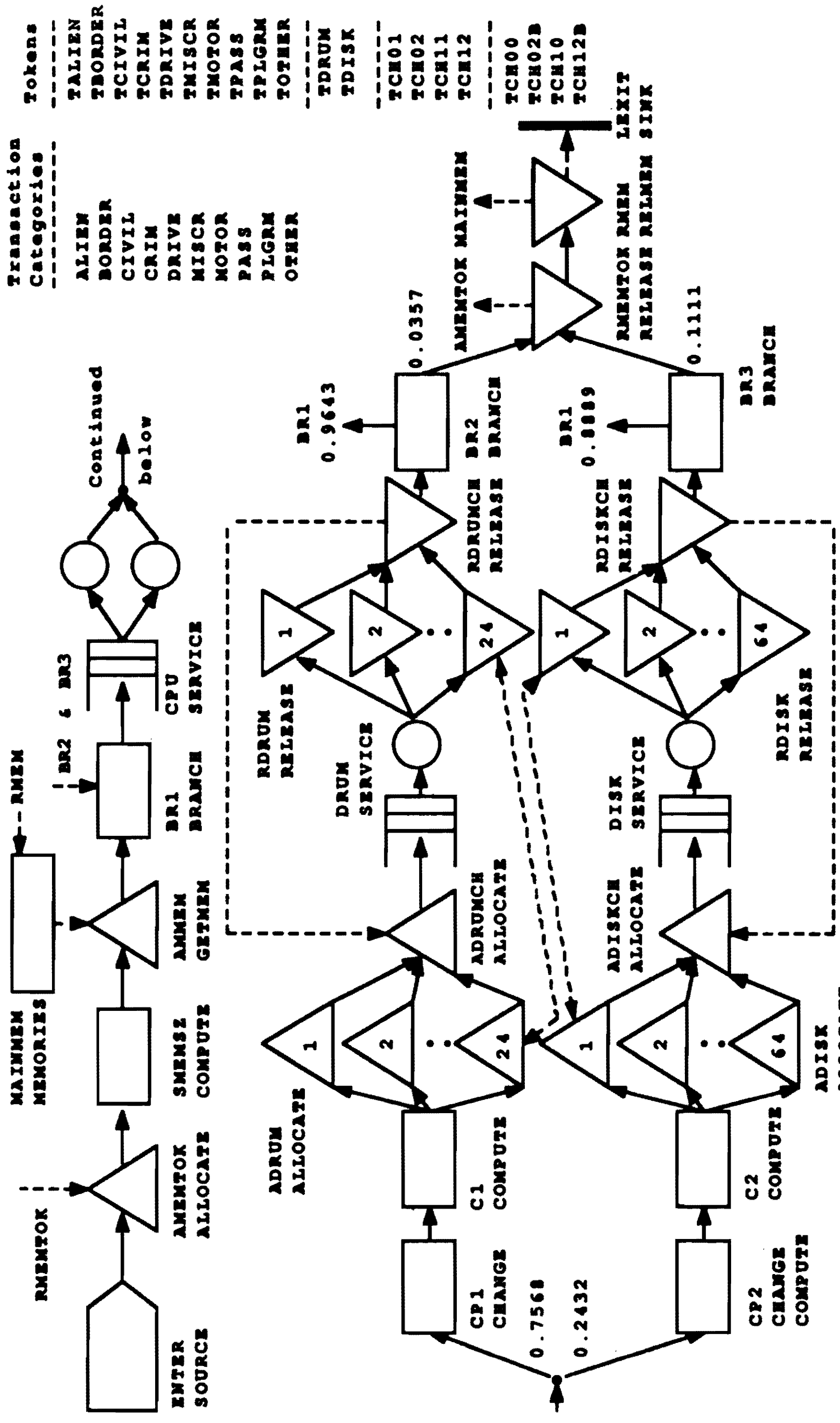

Figure 1. IPG of Drum & Disk Model: SAMIS Online System.

order to route the transaction either into the disk or the drum subsystems.

Each drum string has two control units connected to separate channel interfaces, therefore up to six concurrent drum I/O operations can be carried out. The drum-to-interface connections have been built into the model.

A transaction is routed to one of the 24 drums in proportion to the free space available on that drum, irrespective of transaction category, irrespective of file distributions on drums, and also irrespective of any other CSTS-II activity on any drum. The proportion rule is simple. The larger the size of the data allocated on a drum, the higher the probability of accessing that drum.

Drum service time is approximated to be a simple delay equivalent to the average access time. Data transfer time is ignored considering the speed of the drums.

There are two disk subsystems, two storage control units per subsystem, and four strings of disk drives per subsystem. Each string has two channel adapter units connected to different storage control units to enable simultaneous access to two different drives in the same string. This means there is a total of 64 drives, consisting of 8 strings, therefore up to 4 concurrent disk I/O operations are possible.

A transaction is randomly routed to one of the 64 disk drives since the NIC policy was to distribute all Online System files on disk drives as uniformly as possible.

It is assumed that disk access times are also uniformly distributed between 18.3 to 83.3 milliseconds, *i.e.* between minimum seek time plus latency to maximum seek time plus latency. Disk data transfer time is ignored.

The disk–drum model is unable to represent both the detailed structure of the transactions of the Online Application System and the interdependencies among the transactions, although it models all hardware properties plus the global behavior of the application and operating systems. To remedy the problems associated with this model, it was decided to develop what is called a data-driven model.

The main idea behind the data-driven model is to imitate the flow of a transaction of a function, in terms of task executions and drum and disk I/O operations. Functions consist of multiple tasks, each identified by a stepcode. Tasks initiate each other in a data dependent manner and perform I/O operations on pre-defined files.

The PDL (Program Description Language) description of each function was studied to understand the logic behind task sequences. The task execution sequence of each function was represented by a probabilistic graph, where the nodes of the graph correspond to tasks of a function, edges represent the task precedence relationship, and a probability assigned on an edge represents the probability of the execution of a task after the execution of a previous task. Probabilities are assigned by studying the meaning of the actual function carried out by each task. The probabilistic graph of a function is kept in a data file to be input later to the simulator [5].

The logic in the PDL of each function was further studied to determine the I/O pattern of each task of each function. A data file was prepared to contain information about the identity of the files that are accessed by each task, and how many times a task accesses a specific file.

The Information Processing Graph (IPG) of the data driven model of the SAMIS Online System is shown in Figure 2. The most important node in the graph is the user node. Detailed description of the user node could be found in [3, 4].

The user node in the PAWS IPG contains a number of tables to store information about all data-dependent aspects of the model. These tables are used to assign a phase to a PAWS transaction at the user node, so that the transaction can be routed to execute on a CPU, or perform a drum–disk I/O, or allocate memory, *etc.*

The Task Information Table stores all the information regarding the state, size, I/O pattern, execution time, *etc.* of a SAMIS Online task. FP and FS tables are used to locate the first task and the probabilistic task execution pattern of each SAMIS Online function. The information contained in these tables is as follows.

FP Table : Function pointer table

FP1 :	Function id, *xxyy*
FP2 :	Pointer to the first task of this function in FS

FS Table : Function step indicator table

FS1 :	Stepcode of a function task. If it is zero the task stepcode is the same as the one that has the closest lower task index. This is used

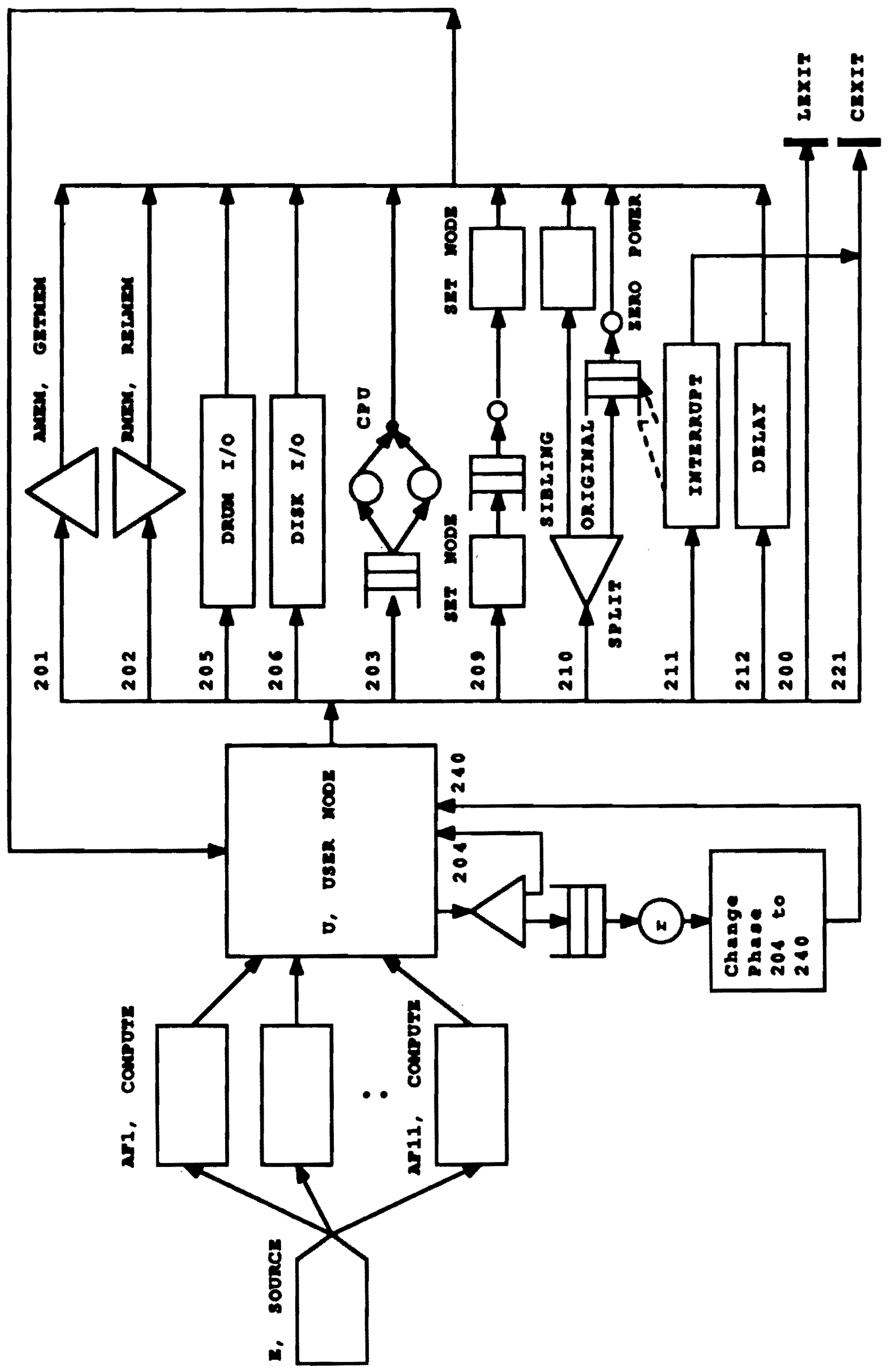

Figure 2. Data-Driven Model.

to define branches in the task sequence diagrams.

FS2	: Pointer to table T to locate the task of this task, FS1(I), of current function
FS3	: Probability of branching to the task pointed by FS4(I). Probabilities are multiplied by 100 and stored as integer values. Probabilities must sum up to 100.
FS4	: Pointer to next task in this function. Zero pointer ends the definition of a function. Next task's index may be higher or lower than current task index.

T Table : Task Information Table
Every column of T is defined as a separate vector variable. Ti refers to column *i*.

T1	: Task id no, *xxdddd*
T2	: Task step code, *dddd*
T3	: I-Bank (Instruction Memory) size in pages
T4	: D-Bank (Data Memory) size in pages
T5	: Flag for task auto-creation at system start-up 0 = Not created, 1 = Created
T6	: Number of waiting terminals, after which another copy of the task will be created
T7	: Maximum number of copies allowed
T8	: Task kill time 1 : if an originally created task (except auto-created tasks) does not execute within this time, it will be killed
T9	: Task kill time 2 : as above, but for task copies
T10	: Number of existing task copies
T11	: State and memory address of each copy of this task given in pairs, up to 3 copies state of *i*'th copy of task *j* is in T11 (*j*, *i*∗2−1) State values are: 0 : In main memory and not busy 1 : In main memory and busy 2 : Swapped out 3 : Being swapped out 4 : Being swapped in 5 : Being created
T11SZ	: Size of T11
T12	: Number of transactions waiting on this task
T13	: Queue to store transaction id numbers of up to 5 waiting transactions *i*'th transaction id for task *j* in T13(*j*, *i*)
T13SZ	: Size of T13
T14	: Task maximum execution time
T15	: Not used now
T16	: Not used now
T17	: Not used now
T18	: Information about files accessed by a task maximum of 10 files.

Transaction flow in the IPG of the data-driven model can be summarized as follows.

Transactions are created according to the computed transaction load. Since the frequency of transactions for a specific SAMIS Online function varies, transactions are assigned a PAWS category according to the computed frequency in nodes AF1 to AF11. Transactions next move into the PAWS user node, in which all operations such as task execution, drum I/O, disk I/O, terminal I/O, *etc.*, will be scheduled according to the function and task information tables previously defined. Transactions are routed out of the user node with a different phase, which identifies the next operation to be carried out by the transaction. Details of nodes are given in [4].

6. RESULTS AND CONCLUSION

Most time and effort during the first twelve months of the project have been spent on finalizing the initially conceived disk–drum model of the SAMIS Online System, and on obtaining crudely approximated results on the performance of the computer system. The results obtained implied that the Online System was capable of providing reasonable service at year 1990 peak load for a certain set of assumptions, but could not provide reasonable service for another set of assumptions. These assumptions were mainly concerned with a few system parameters such as system load in terms of transactions per second, and CPU time distribution. Results that showed reasonable service were obviously in conflict with the service rates that were being experienced by the users of the system under a fraction of the projected peak load.

A large number of simulation runs of the disk–drum model in PAWS were conducted. Each simulation run took a considerable amount of job execution time. Simulation run times as much as five to six hours of VAX-11/780 time were experienced even when the current model run on the VAX system while no other users were sharing it. The PAWS software package configuration used in simulation runs required approximately four megabytes of virtual main memory, and our VAX system could supply at most only around 1.2 megabytes of real memory if the package was run alone in the system. Unacceptably long run times were experienced due to excessive paging activity because of the small real memory.

A related problem was experienced while the PAWS package was being configured to run the model. Because of the structure of the PAWS package, it is possible that the package will simply abort a simulation run which has already taken a considerable amount of time, due to overflows in PAWS internal tables. Such overflows happen to show up one by one and especially towards the end of the simulation run. Each abort results in the update of one or more PAWS configuration parameters, then the compilation and linking of all package modules to recreate a new package with more capacity, which are themselves quite long jobs, runs on the VAX system. The PAWS package was re-configured about twenty times during the last six months, resulting in the loss of considerable machine time and human effort.

The selection of the model simulation time very much affects the resulting package run time. The actual simulation time is restricted to 240 000 ms, or four minutes. It seems there is no necessity to simulate the model for a duration equivalent to approximate the SAMIS daily up time, perhaps 8 hours, during which a reasonable load is handled. Since the simulation runs are made by using roughly the peak system load, actual short simulation times will be acceptable as long as it is demonstrated that there is no accumulation of unprocessed transactions left in the model when the simulation time is over. All simulation runs carried out satisfied this last criteria.

System modeling scenarios consisted of an examination of the effects of main memory size, drum channel bandwidth, drum access time, disk channel bandwidth, and disk access time on response time distribution, CPU utilization, memory utilization, and drum–disk utilizations.

Simulation runs with different input transaction rates and different mean CPU times were conducted. The results obtained for a set of assumptions imply that the SAMIS Online System, as modeled, is capable of handling the expected peak load well, in fact, with an average response time and standard deviation less than 0.5 seconds, or 500 milliseconds, with reasonable utilization of all hardware resources. This looked quite good compared with what is actually happening in the SAMIS Online System. The response time increases very rapidly with increase in the transaction rate. High transaction rates are expected to result when incoming transactions create a number of transactions in other transaction categories. This action was called transaction dependency.

The disk–drum model was not capable of representing the fine details of the transactions of the application software and its components. Therefore, an additional data-driven model was constructed to represent the finer details and to extract more reliable and precise performance measures. Most of the work done to construct the disk–drum model was also used in the data-driven model; therefore the effort spent on developing the disk–drum model was not wasted at all.

In the data-driven model, only the data regarding the complete specification of the Citizen Registry subsystem were extracted and incorporated into the model. Most of the effort was spent on designing and coding the central component of the data driven model, called the user node. The information extracted included properties of all SAMIS Online System files, the task structure of SAMIS Online System functions, properties of all system and application tasks, and task-file relationships.

Because of the excessive time required to extract data for the remaining subsystems, only the correctness of the data-driven model was verified without actually creating any performance evaluation statistics. Some hypothetical results obtained by projecting the Citizen Registry subsystem load into full transaction load are documented in [5] to understand the behavior of the model, but these results do not represent the actual system performance.

After the task and file information data of the remaining subsystems are extracted from the SAMIS Online System documentation and built into the model, it would be possible to have meaningful performance statistics about the modeled system.

ACKNOWLEDGEMENTS

The author would like to thank Dr. A. N. Sagr, Director General of National Information Center, Ministry of Interior for enabling and supporting the performance modeling and evaluation task of the project, and Dr. M. I. Al-Suwaiyel, Dean, College of Computer Science and Engineering, KFUPM, Dhahran for heading both tasks of the project. Many thanks are also due to Dr. E. T. Upchurch for initiating the disk–drum model, P. J. Parmar of CSC for his cooperation in providing information and documentation on SAMIS computer network, and K. W. Al-Dhaher for his effort to complete many simulation runs.

The work described in this paper has been carried out while the author was with the Information and Computer Science Department, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia, and has been fully supported by National Information Center, Ministry of Interior, Riyadh, Saudi Arabia, through Saudi Arabian National Center for Science and Technology.

REFERENCES

[1] M. I. Al-Suwaiyel, S. S. Hyder, and M. U. Caglayan, "SAMIS Computer Network: Evaluation and Enhancements", *Progress Report I, Computer and Information Science Department, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia,* September 1985.

[2] M. I. Al-Suwaiyel, S. S. Hyder, and M. U. Caglayan, "SAMIS Computer Network: Evaluation and Enhancements", *Progress Report II, Computer and Information Science Department, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia,* April 1986.

[3] M. I. Al-Suwaiyel, S. S. Hyder, and M. U. Caglayan, "SAMIS Computer Network: Evaluation and Enhancements", *Progress Report III, Computer and Information Science Department, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia,* September 1986.

[4] M. I. Al-Suwaiyel, S. S. Hyder, and M. U. Caglayan, "SAMIS Computer Network: Evaluation and Enhancements", *Final Report, Computer and Information Science Department, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia,* December 1986.

[5] K. W. Al-Dhaher, "Performance Modeling of a Computer System: Saudi Arabian Ministry of Interior Online System", *M.S. Thesis, Computer and Information Science Department, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia,* June 1987.

[6] *An Introduction to PAWS.* Austin, Texas: Information Research Associates, 1984.

[7] *PAWS 2.0 Users Manual.* Austin, Texas: Information Research Associates, 1984.

[8] C. H. Sauer and K. M. Chandy, *Computer Systems Performance Modeling.* Englewood Cliffs, NJ.: Prentice–Hall, 1981.

Paper Received 13 July 1988; Revised 11 March 1990.