Information Processing in Japan Vol. 11, 1971

PACSS——A Simulator for Performance Evaluation of Computer Systems

TORU MIKAMI*, HIDEHITO KUBO*, ISAO TAKAHASHI*,
YOSHINORI ARIFUKU* AND TAKASHI KITAURA**

Abstract

A general model of interactive computer systems is presented. It is shown that the overall action of a computer system which includes terminal users, user processes, an operating system and a hardware system can be modelled by mutual actions of a set of four substantive and three functional model elements.

A computer system simulator PACSS based on this model is also presented. It provides a block-oriented language system and a characteristic reporting facility.

1. *Introduction*

One of the most important problems in the development of modern complex computer systems, such as time-sharing systems, is the evaluation of system performance.

There are two approaches to this problem. One is the approach based on the analytical theory, such as queuing theory or Markov process theory. The other is the method which utilizes the computer simulation technique. Since system models which can be solved by the analytical method are restricted to very simple ones in its present ability, we can hardly expect this method to be of use for exact performance evaluation of computer systems. By the simulation method, on the contrary, we can analyze any system model in any degree of preciseness though the solutions obtained by this method are not so clear as that obtained by the analytical method. For this reason, the simulation method can be considered a promising means for performance evaluation of computer systems.

We must next consider whether we should use the general purpose simulation tool, such as GPSS or SIMSCRIPT, or whether we should develop a new problem-oriented simulation tool. Generally, in order that we can effectively perform a simulation analysis, it is required that the simulation tool used in it should fulfil the following conditions:

This paper first appeared in Japanese in Joho Shori (the Journal of the Information Processing Society of Japan), Vol. 12, No. 1 (1971), pp. 14-25.

* Cetral Research Laboratories, Nippon Electric Co., Ltd.
** Data Communication Systems Division, Nippon Electric Co., Ltd.

(1) The correct modelling of objective systems is possible and the model can be described with ease. Parameterized description forms, if possible, are desirable.
(2) The flexible modelling is possible.
(3) The debugging can be performed with ease in a short time.
(4) The run time ratio defined as

$$\left[\frac{\text{the actual time required for model execution}}{\text{the simulated elapsed time during model execution}}\right]$$

is as low as possible.
(5) Necessary and sufficient statistical outputs can be generated in answer to observation requirements.

Usually, parameter changes, model modifications and other simulation condition changes are required very often during a process of simulation analysis. Therefore, among the requirements described above, (1) is most important, as was pointed out by Heusmann et al [1]. In addition, the abundance and variety of generated statistics are very desirable for our purpose, the evaluation of complex systems like time-sharing systems. For these reasons, it is required to develop a new problem-oriented simulation tool for computer system evaluation.

PACSS (Program And Computer System Simulator) was developed by the authors in response to this requirement and it has been used for several simulation studies. The purpose of this paper is to present a computer system model which forms the theoretical basis of PACSS. Also shown is an outline of PACSS which is an implementation of this computer system model.

CSS of IBM and S3 of CEIR are typical computer system simulators which were developed with the object similar to the authors' [2], [3].

2. *Computer System Model*

2.1 *The Concept of Modelling*

Let us consider a man-computer-interactive system which consists of four components:

(1) A set of terminal users who generate jobs.
(2) A set of user job processes which perform generated jobs.
(3) An operating system which controls these activities and manages resources.
(4) A hardware system which realizes all these activities.

We call the model of such system a computer system model (CS model).

In order to clarify the structure of CS model, we must consider it from two standpoints. When we see CS model from the substantive viewpoint, it can be regarded as a set of elementary entity models which can be obtained by the abstraction of elementary entities of the real system. On the other hand, if we grasp it from the functional point of view, it can be seen as a relation of ele-

mentary functional models which can be obtained by abstracting elementary functions of the real system.

Let us introduce four substantive notion in order to define CS model.

Process transaction: An abstracted entity which represents a user job process to be executed by a processor.

Message transaction: An abstracted entity which represents a message that transmits a unit of data or an order.

Activator: An abstracted entity which represents the control of a prosessing unit.

Resource: An abstracted entity which represents a resource. This entity is classified into three types, facility, storage and peripheral resource.

In addition, let us introduce three functional concepts.

Job process: An abstracted function which represents activities of a user job process.

Job generation: An abstracted function which represents activities of a terminal user.

Activity block network: An abstracted function which represents the control operation of an operating system. Necessary hardware operations are imbedded in this function.

By using entities introduced previously, structures of these functions are defined as follows:

Def. 1: Job process is a finite sequence of pairs (S, E), i.e.

$$(S, E)_1, (S, E)_2, \cdots\cdots, (S, E)_m,$$

where E denotes an event of generation of a mesage transaction and S denotes a net processing time in the interval between two adjacent E occurrences.

Def. 2: Job generation is a finite sequence of pairs (T, F), i.e.

$$(T, F)_1, (T, F)_2, \cdots\cdots, (T, F)_n,$$

where F denotes an event of generation of a message transaction and T denotes an interval from the time when a generated message transaction terminates to the occurrence of the next F.

Def. 3: Activity block network is a directed graph which is obtained by connecting a set of activity blocks with a set of branches according to a relation, where an activity block is a representation of an element of functions performed by activators, message transactions and process transactions.

Performing the functional element represented by an activity block is realized in a way that each of activators, message transactions and process transactions passes the activity block and executes the function defined by the activity block. Activators, message transactions and process transactions flow on the activity

block network and perform the function defined by the network.

Now, by using the elementary concepts defined above, we can give a definition to CS model.

Def. 4: CS model is a 7-tuple

$(A, M, P, R; J, G, B)$,

where A denotes a set of activators, M denotes a set of message transactions, P denotes a set of process transactions, R denotes a set of resources, J denotes a set of job processes, G denotes a set of job generations and B denotes an activity block network. J, G and B define relations of A, M, P and R, and define activities to be performed by A, M, P and R.

In CS model, J is related to B by means of mutual transfers of message and process transactions. Also, G is related to B by means of mutual transfers of message transactions. Message and process transactions flow on B and perform functions defined by B. Relations of these activities are shown in Fig. 1.

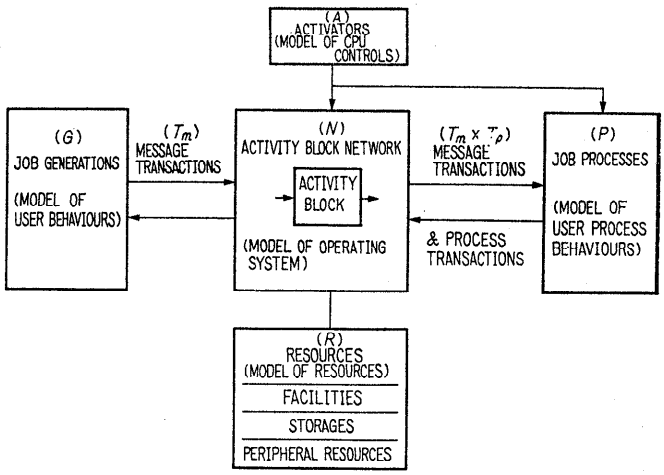

Fig. 1. The conceptual block diagram of CS model.

2.2 *Job Process*

The job process simulates the behaviour of the user job process in the computer. The behaviour of a user job process is characterized by three parameters shown below (see Fig. 2):

(1) A sequence of supervisor call (SVC) occurrences (① in Fig. 2).

(2) A sequence of intervals between two adjacent SVC occurrences (②).

(3) A total time or total number of SVC occurrences necessary for finishing the process (③).

The parameter (1) specifies the sequence of E's of the job process. The SVC sequence can be expressed in the nullth order or first order Markov chain of SVC's. Or it can be expressed in the nullth order or first order Markov chain

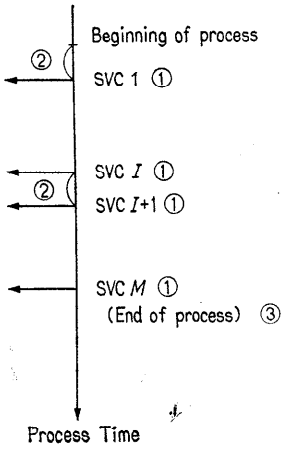

Fig. 2. The model of user job process.

of deterministic SVC subsequences having arbitrary lengthes.

The parameter (2) specifies the sequence of S's. Here the SVC interval means a net required time for processing a user job process between two SVC occurrences.

The parameter (3) specifies the length of (S, E) sequence.

2.3 *Job Generation*

The job generation simulates the behaviour of the terminal user. The terminal user activities can be considered to be composed of a sequence of conversations each of which consists of a sequence of command jobs. Each command job may include interactive I/O's or may not.

Therefore, in the job generation, the operating characteristics of a terminal user are defined by three parameters shown below (see Fig. 3):

(1) Intervals between two adjacent conversations (① in Fig. 3).

(2) A sequence of command occurrences in each conversation (②).

(3) Intervals between two adjacent command jobs and intervals from interactive output arrivals to the next input occurrences in each command job (③).

The parameter (2) and the sequence of interactive input occurrences specify the sequence of F's of the job generation. The command sequence can be expressed in the nullth or first order Markov chain of commands. Or it can be expressed in the nullth or first order Markov chain of deterministic command subsequences having arbitrary lengthes. The sequence of interactive input occurrences is not given explicitly. It is automatically generated in response to occurrences of SVC's in the job process which require interactions.

The parameter (1) and (3) specify the sequence of T's. The interval defined in the above (1) and (3) is defined more exactly as a time from the instant when an output from a computer to a terminal begins until the instant when the next

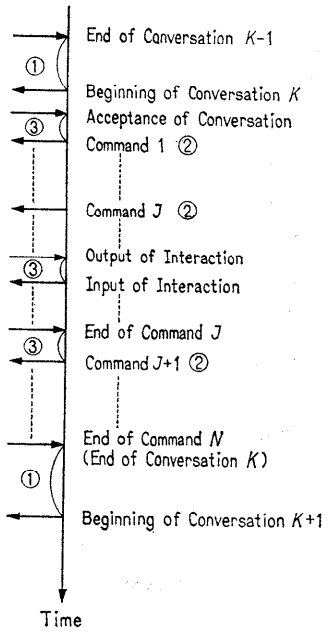

Fig. 3. The model of user job generation.

input from the terminal to the computer ends. Hence, the interval consists of output time, thinking time and input time.

The lenth of (T, F) sequence is not specified explicitly. It is determined by the simulation interval on its execution.

3. *PACSS*

3.1 *PACSS Program*

PACSS is a computer system simulator which is an implementation of CS model. PACSS consists of three subsystems, Editor, Simulation Executor and Report Generator. The major parts of these programs are written in BPL (Basic Programming Language : a subset of PL/1 developed in Nippon Electric Company). The total size of these programs is approximately 38,000 statements in BPL plus 300 steps in an assembly language.

Each program can be executed on NEAC series 2200 computers having more than 256 k character memory capacity.

3.2 *Event*

PACSS does not simulate the system at each successive interval of time. It updates the clock to the time at which the next most imminent event is to occur.

Three kinds of events are defined in PACSS:

PU event: An event at which a transaction is due to depart an activity block.

External event: An event at which an external interruption to the block activity is due to occur.

System event: An event at which an action defined by the system (PACSS) is to be performed.

Typical system events are the start of measurement, a sampling of state variables and the suspension of simulation execusion.

3.3 *Block-oriented Language*

PACSS provides a block-oriented language system for describing CS models. The main body of the language system consists of twenty-five standard activity blocks and five measurement blocks. Standard activity blocks are used for describing activity block networks, job processes and job generations. Measurement blocks are used to obtain specific statistical outputs and to aid model debugging.

PACSS language system contains such blocks as are shown below.

- ○ A block which executes the function of job process.
- ○ A block which executes the function of job generation.
- ○ Blocks for handling transactions (three standard blocks).
- ○ Blocks for seizing or releasing resources (four standard blocks).
- ○ Blocks for varying values of system variables (two standard blocks).
- ○ Blocks for treating external events (four standard blocks).
- ○ Blocks for handling transaction queues (four standard blocks).
- ○ Blocks for branching routes of transaction flows (three standard blocks).
- ○ A block to which any function can be given by coding.
- ○ Blocks for measuring system variables (three measurement blocks).
- ○ Blocks for tracing motions of transactions and activators (two measurement blocks).

3.4 *Statistical Outputs*

Statistical outputs of PACSS contain such items as are shown below:

- ○ Statistics of response times by every command and/or for all commands.
- ○ Statistics of conversation times and command times.
- ○ The total of passing times of blocks belonging to each class. The totals represent, for example, net used times, overhead times and idle times of processing units.
- ○ Statistics of facility, storage and peripheral resource usages.
- ○ Statistics of transaction queue lengthes and waiting times.
- ○ Statistics of sampled variables and specified system variables.
- ○ Edited trace data of transactions.

3.5 *The Simulation Mechanism*

The simulation execution of system models which are described in PACSS language and translated into internal codes by Editor is carried out by Simulation Executor. Fig. 4 shows an outline of simulation mechanism of Simulation Executor.

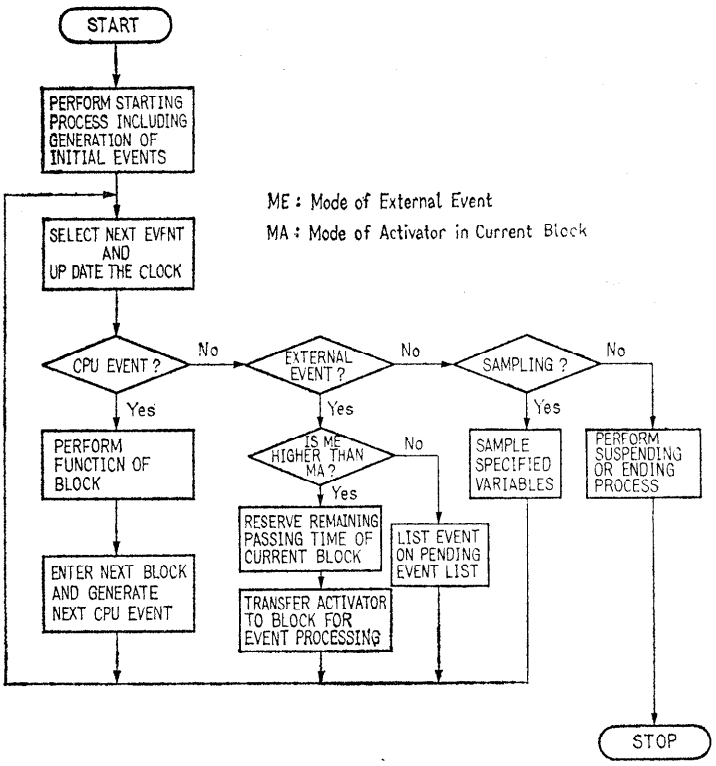

Fig. 4. General flow chart of simulation executor.

4. *Conclusions*

There have been shown a computer system model (CS model), an outline of a computer system simulator PACSS based on CS model.

In CS model, the whole structure and action of a computer system can be represented by a 7-tuple of four substantive model elements and three functional model elements. The formers are process transactions, message transactions, activators and resources. The latters are job processes, job generations and an activity block network.

PACSS provides a block-oriented language system for describing CS models and a reporting facility for generating statistical outputs peculiar to computer system performance evaluation.

Acknowledgement

The authors wish to express their gratitude for the encouragement received from Dr. H. Watanabe during this work. Acknowledgement is due to Mr. K. Ogata for his valuable suggestion.

References

[1] Huesmann, L. R. and R. P. Goldberg: "Evaluting computer systems through simulation", *The Computer Journal*, August, pp. 150–156 (1967).
[2] Seaman, P. H. and R. C. Soucy: "Simulating operating systems", *IBM Systems Journal*, Vol. 8, No. 4, pp. 264–279 (1969).
[3] Cohen, L. J.: "System and software simulator, S3", CEIR Inc., 5272 River Road, Washington, D. C. (1968).