AN ANALYTICAL MODEL OF A TELEPROCESSING SYSTEM

Terence Berinato
Ætna Life & Casualty Corp.
Hartford, Connecticut

ABSTRACT

A queuing model has been developed to study the performance and capacity of a casualty insurance teleprocessing system. This paper presents the salient features of the system itself, relates those features to basic queuing theory algorithms, outlines the basic model construction, and discusses the validation results.

INTRODUCTION

At Ætna Life and Casualty we use a large scale teleprocessing system to process our casualty insurance messages. The computer consists of an IBM/370-168 CPU using MVT and TCAM. The message processing program, called the REX, uses a master task to interface with TCAM, and several lower priority subtasks to control the path the message takes through the processing programs. An overview of the system is shown in Figure 1. TCAM classifies the messages as:

-SAFA - a high priority conversational mode message

-SAFB - a low priority data entry mode message.

The processing program classifies the messages by type of processing as:

- Policy Change
- Inquiry
- New Business
- Cancellation
- Renewal
- Claim.

The master task can schedule any subtask to handle any message, i.e., there is no unique association of message type and subtask. Messages arrive at the CPU over the TP lines and wait in the TCAM Input Queue buffer.

TCAM picks the messages up, processes them and puts them in a Queue for the Master Task (all SAFA ahead of all SAFB). The master task waits for a free subtask and schedules a message. If there are no SAFA or SAFB messages, the master task waits $\frac{1}{2}$ second and rechecks the queues. When the subtask finishes processing a message, it passes it back to TCAM (if it's SAFA), or stores it on a disk file (if it's SAFB). The overall times measured by the system are: 1) arrival at the CPU until process program pick up (called QIRI), 2) process program time (called RIRO), 3) and arrival at the CPU until process program completion (QIRO). In addition, resource utilization (i.e., CPU, Channels devices) are measured by CUE. (Boole & Babage Software Monitor).

MODEL OVERVIEW

Messages arrive at the CPU ar the rate of λ TP (see figure 2) and wait in the TCAM input queue. TCAM (an infinitely numbered software server) processes them and puts them in the process program input queue as λ SAFA and λ SAFB. As a first approximation, the process program master - subtask system is treated as a machine interference problem. Thus, the average wait of a message at the TCAM - process program interface will be function of the relative arrival rates of SAFA vs. SAFB, and the probability of a free subtask. The model timings of interest at this level will be the same ones measured by the system QI-RI, RIRO, and QIRO.

MODEL DETAIL

To find the overall timings and time distributions of both of the major components, each must first be broken up into a detailed network of smaller, simpler, queuing systems. Each node of the network will have an exponential arrival rate and general service time distribution. Then each node can be analyzed independently using the appropriate queuing theory formulas. After the timings of all nodes are found, they can be summed to get the average service time and time distribution functions of the major components. One major potential problem with this method is that general service time distributions of some nodes cause non-random output rate distributions. These output steams become input streams to successive nodes. Queuing theory formulas do not accurately describe systems with generalized input rate distributions. However there are so many interconnections in the network on the detailed level that, the net result of all the input streams to any node approximates a random arrival rate.

On this detailed level there are only 8 basic servers. They are, the CPU, 6 Block Multiplexer

channels with about 32 disk drives each and a tape selector channel (Fig. 3). The CPU was treated as a single server preempt - resume priority system with the operating system, TCAM, and the process program being the 3 priority levels. Arrivals to the CPU consist of SVC's to the operating system, and processing requests generated by message arrival rates at CPU nodes of TCAM and the application program. The service timings for these arrivals were determined from documentation, measurements, and our knowledge of system operation.

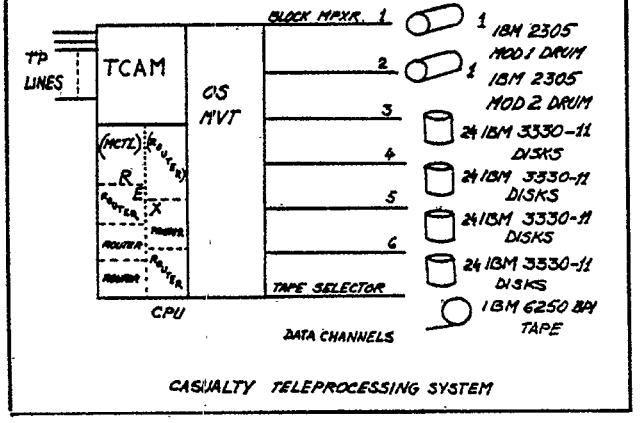
Each Channel device system was a single server machine interference system. Arrivals were the requests for I/O. Service times and distribution were based upon hardware characteristics. The specific layout of files on devices and channels was determined from system documentation.

In the real system, program logic determines when the CPU is used, when I/O is done and when to branch. Similarly, the interconnections among the CPU and I/O nodes in the model is controlled by descriptions of the processing programs and TCAM.

The model was developed because the problem is too complex to work by hand. A simpler version of this evaluation can be done by hand (Ref. No. 1, Example 32), but a simpler model will not be accurate enough to answer the questions asked about the system. The computer model finds the arrival rates at the nodes, calculates utilizations (CPU, Channel and Device), evaluates the timings, and finally sums the timings and variances to predict major component timing. Figure 4 is a flow chart of the model.

The model used several large arrays to define the program logic, message mix, and file and device locations. The arrays were cross indexed so intermediate results did not need to be duplicated. Most data is input to the model at execution time. Thus many parameters including message mix, program logic, file locations, device characteristics, and OS interrupt processing are treated as variables and easily changed.

FIGURE 1.



After reading in input data about device and channel characteristics, file and record characteristics, and OS interrupt characteristics, the program reads in the process threads of each message type (e.g., the sequence of file accessing, CPU utilization, and percent branching). Then, the file accessing is broken out by file name, and the average CPU time between each file access is identified . Also, the arrival rate to each of these nodes is identified in terms of message type, percent error and percent repeating each step. The program then determines the CPU utilization (ρ CPU), channel utilizations (ρ CHL), and device utilizations (ρ DEV). As mentioned earlier, the CPU is a single server preempt resume priority system, and the channels and devices single server machine interference systems. To get the CPU supervisor utilization due to I/O requests the arrival rates are summed by file name across all message processing paths.

$$\text{i.e.,}\quad \lambda_{\text{file}} = \sum_{k=\text{paths}}^{\text{path m}} \sum_{j\ \text{steps}}^{\text{step m}} \lambda\ \text{file jk.}$$

Then the file arrival rate is distributed across all devices on which the file resides. By summing all the I/O references and other supervisor requests (e.g., POST, WAIT, EOT, END-I/O) the program finds the number of requests for CPU service per second. From a knowledge of the CPU time per CPU interrupt type, the supervisor state CPU utilization, the average and 2nd moment service time can be found. The second and third priority levels are TCAM services and application CPU times, and they are found by summing the CPU for each step.

Thus, $\rho_{\text{CPU}} = \rho$ CPU OS + ρ CPU TCAM + ρ CPU REX. ρ's, T_{B_2} for each priority level allows calculation of Tw and Tq, the wait and total time for each CPU request at each priority level.

From the I/O interrupt portion of the CPU calculation above, the program knows the number of accesses/second (λ i) to each device. From all the input information the model finds ρ channel, ρ device, Ts channel, Ts Device. Assuming each channel and its devices to be a machine interference problem, the TW_{CH} and TW_{DEV} can be found for each device.

FIGURE 2.

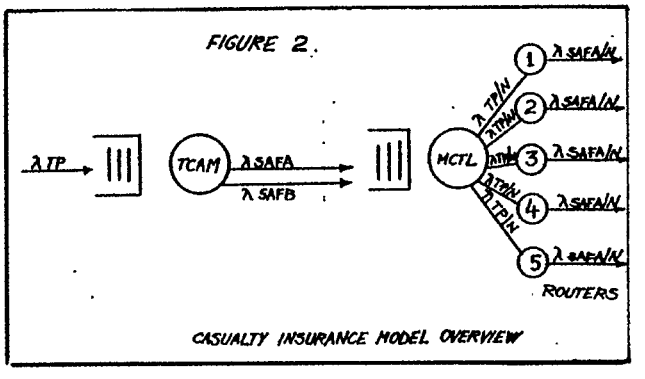
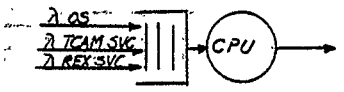
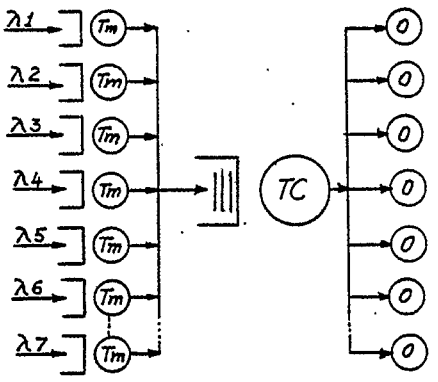


FIGURE 3.



CPU SERVER
PRE-EMPT RESUME PRIORITY
SINGLE SERVER



DISK-CHANNEL MACHINE MODEL
Tm = DISK SEEK TIME
TC = CHANNEL SERVICE TIME

The utilizations calculated in each of these steps are useful in themselves since they are independent of any queuing theory assumptions. They provide an accurate view of system capacity even if the timings calculated using the queuing theory assumptions are wrong. Practical experience gives us an idea when utilizations are too high. Thus at this step the model already provides useful information.

Next, proceeding backwards, the program sums the average and second moments of wait and service time across all devices on which a file is located giving the averaging and variance of the time to access any file.

Lastly, these CPU and I/O times can now be summed down each program processing path to give the average and variance of the service time of the software servers in Figure 2. By applying the appropriate queuing theory formulas to each of those major servers the overall times can be found.

DISCUSSION OF RESULTS

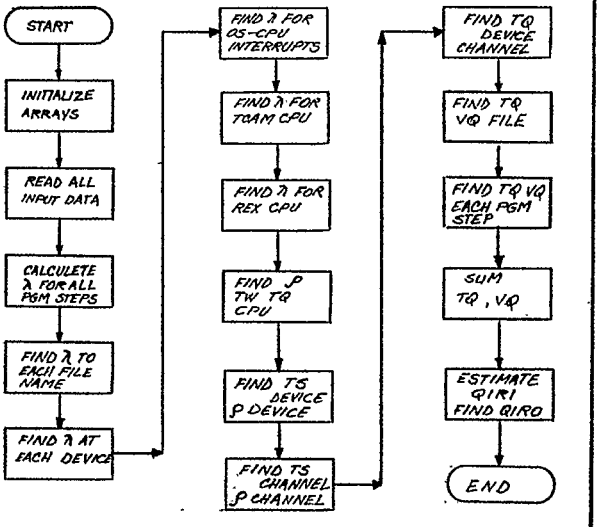
Tables 6 and 7 compare model and measured system results. The most impressive feature of the model is the accuracy of the CPU & Channel utilizations. If nothing else were correct this output would justify the cost of the model. Granted, they are the product of the mechanical summing of processing paths, but they are still quite useful for performance prediction, and they help verify the accuracy of the model.

Of the calculated timings, the most accurate is RIRO. This timing was the sum of the average timing in the heart of the network where the approximations to random arrivals was most accurate. The QIRI was the least accurate timing in the model. This was due to the difficulty in modeling the combination of timer delay, priority of arrivals and non-random output from TCAM. I attempted to use a weighted probability of the master task using a timer based upon the probabilities of a free subtask, RIRO times and probability of an arrival based on random arrivals. This error accounts for the error in QIRI times as well.

However, and the most important from my viewpoint is the accuracy of these results relative to the cost of acquiring them. We have a detailed discrete event model of the same system whose accuracy is maintained at ± 10% of system measurements. Except for QIRI, most results of the analytical model are of comparable accuracy but the analytical model costs one fortieth as much to run. So on a cost-performance basis this model should be considered a success.

Finally, most questions which arise about the system performance and hardware selection are so obvious that no model is needed at all. Of the remaining questions, if the answers are so close that the model's error becomes a factor (i.e., ± 10%), then one usually moves on the side of caution. This model is a useful tool and much cheaper to use than our discrete event model. Thus while discrete event models definitely have a place, this project convinces me they should be only considered as a method of last resort.

FIGURE 4.



ANALYTICAL FORTRAN MODEL FLOW

TABLE 5

PRINCIPLE MODEL ARRAYS

Array	Description	Input Items
Program Step	Contains a description of each node of a program. Including: Type (CPU, IO, Branch) Value (MS, File No, %) MSG Arrival rate Avg time (wait plus service) Variance of (wait plus service)	Type Value
Message Mix	Contains the percentage of each message type.	All items
File Names	Contains File wide characteristics. e.g., File ID Nr. record length, arrival rate of I/O requests. Average I/O time. Variance of I/O time.	File ID No. record length
File Device Association	File ID No. Channel Nr. Device Nr. on the channel amount of the file on the device (e.g., No. of cylinders) I/O requests to this File-Channel-Devices (FCD) Average time to access this FCD Second Moment of " to "	File ID Nr. Channel Nr. Device Nr.
Device Characteristics	Device No. Seek time Latency Transmission rate	All items

FIGURE 6

SYSTEM MEASUREMENTS

SAFA TRX/MIN	26.6	31.8	37.0	37.6	40.2
SAFB TRX/Min	74.2	84.1	97.4	98.7	100.5
TOTAL TRX/MIN	100.8	115.9	124.4	136.3	140.7
CPU TOTAL UTIL %	48.6	52.3	56.8	65.8	65.0
" SUPRV UTIL %	10.6	12.3	13.3	14.3	15.0
" REX UTIL %	21.8	27.0	28.2	32.4	33.5
" TCAM UTIL %	15.1	12.0	14.0	17.5	15.0
" OTHER UTIL %	1.1	1.0	1.3	1.6	1.5
CHANNEL 1 UTIL %	31.2	38.1	40.8	45.8	46.7
" 2 UTIL %	30.1	39.2	38.1	45.3	45.4
" 3 UTIL %	10.6	12.3	10.1	11.8	10.8
" 4 UTIL %	8.3	7.4	8.4	9.6	9.5
" 5 UTIL %	11.5	11.5	13.4	15.0	14.8
" 6 UTIL %	5.3	6.7	6.8	8.5	8.5
QIRI SAFA (Sec)	0.4	0.4	0.4	0.4	0.5
" SAFB (Sec)	0.8	0.9	1.1	1.4	1.7
RIRO SAFA (Sec)	1.2	1.3	1.3	1.4	1.5
" SAFB (Sec)	1.4	1.5	1.5	1.6	1.6
QIRO SAFA (Sec)	1.6	1.7	1.7	1.8	2.0
" SAFB (Sec)	2.2	2.4	2.6	3.0	3.3

FIGURE 7

MODEL RESULTS

SAFA TRX/MIN	26.6	31.8	37.0	37.6	40.2
SAFB TRX/MIN	74.2	84.1	97.4	98.7	100.5
TOTAL TRX/MIN	100.8	115.9	124.4	136.3	140.7
CPU TOTAL UTIL %	47.0	52.3	55.5	59.5	61.1
" SUPRV UTIL %	9.7	11.2	12.0	13.1	13.5
" REX UTIL %	23.2	26.6	28.7	31.3	32.3
" TCAM UTIL %	14.1	14.5	14.8	15.1	15.3
" OTHER UTIL %	—	—	—	—	—
CHANNEL 1 UTIL %	29.1	33.5	36.1	39.3	40.6
" 2 UTIL %	26.2	30.1	32.5	35.4	36.6
" 3 UTIL %	14.2	16.3	17.7	19.2	19.9
" 4 UTIL %	12.6	14.5	15.7	17.1	17.7
" 5 UTIL %	13.0	15.0	16.0	17.6	18.1
6 UTIL %	4.0	4.6	5.0	5.4	5.6
QIRI SAFA (Sec)	0.7	0.8	0.9	1.0	1.1
" SAFB (Sec)	1.1	1.3	1.7	2.2	2.6
RIRO SAFA (Sec)	1.1	1.2	1.2	1.3	1.3
" FAFB (Sec)	1.1	1.2	1.2	1.3	1.3
QIRO SAFA (Sec)	1.8	2.0	2.1	2.3	2.4
" SAFB (Sec)	2.2	2.5	2.9	3.5	3.9

BIBLIOGRAPHY

1) IBM manual GF20-0007-1, Analysis of Some Queuing Models in Real Time Systems,

2) A. O. Allen, Elements of Queuing Theory for Systems Design, IBM systems Journal Nr. 2 1975

3) Hisashi Kobayaski and Alan G. Konheim, Queuing Models for Computer Communication System Analysis, IEEE Transactions on Communications Vol., Com 25, Nr. 1, January 1977

4) M. Reiser, Interactive Modeling of Computer Systems, IBM Systems Journal, Nr. 4 1976