Contribution to the Tenth International Teletraffic Congress (ITC), June 1983, Montrêal, Canada.

QUEUEING ANALYSIS OF AN ARRIVAL-DRIVEN MESSAGE TRANSFER PROTOCOL

D.R. MANFIELD
P. TRAN-GIA

BELL-NORTHERN RESEARCH, CANADA
UNIVERSITY OF SIEGEN, FRG

ABSTRACT

In SPC switching systems with distributed architecture, the overall performance is affected by the efficiency of the messaging protocols used for inter-processor communication. Since there is processor overhead associated with the interrupt routines used for message passing, it may be better to pass messages in batches. In this paper the performance analysis of a queueing model for a messaging protocol called "collect n with timeout" is presented, in which the major performance measures are derived, along with numerical results to demonstrate the optimising effect of the protocol parameters.

1. INTRODUCTION

Many modern SPC exchanges are constructed with a hierarchical architecture where the concurrent real time call processing being done in different processors is synchronized by the passing of messages. Typically, lower level call processing tasks such as scanning, signalling and supervision are implemented by peripheral decentralized processors, each controlling a group of lines or trunks. Higher level call processing such as resource allocation, network path selection and route selection is done in a central control processor. The messaging system is implemented with signalling processors at the various hierarchical levels, connected by signalling links. The purpose of the signalling processors is to perform the scanning, polling and message transmission functions of the message transfer protocol between the hierarchical levels of the switch. The message traffic is made up of all the call processing and maintenance messages passed between the periphery and the central control.

In contrast to monolithic SPC systems (which need no messaging system), the messaging system of the distributed system constitutes a performance critical component in that messaging delays contribute to the switch grades-of-service (dial tone delay, call set up delay). The performance of the messaging system is determined to a large degree by the efficiency of the transfer protocols used between the distributed processors. The analysis of the transfer protocols may be carried out by treatment at a subsystem level such as depicted in Figure 1.

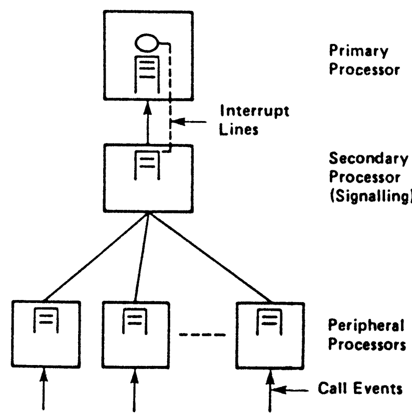

FIGURE 1: SWITCH MESSAGING SUBSYSTEM

Messages representing telephony events generated by the peripheral processors are collected by the Signalling Processor, hereafter referred to as the Secondary Processor (SP), and stored in outgoing message buffers (mail boxes) awaiting transfer to the main processor which we designate the Primary Processor (PP). When the SP wishes to initiate a transfer of messages it sets a flag to generate an interrupt in the PP, which suspends whatever it is doing to run its I/O routines, and store the incoming messages in its work queue for subsequent processing. This procedure involves a certain amount of overhead for the PP and to minimize this overhead it is better to pass the messages in batches. This will involve some extra delay in the outgoing buffer of the SP, but with a possibility of faster processing in the PP.

The level of message traffic in an SPC switch can be very high, so there are significant gains to be made by passing messages in batches. On the other hand, at low traffic levels, it may take too long to collect the required batch, so it is necessary to protect the transfer mechanism by a timeout after which a transfer is initiated anyway.

In this paper we present the performance analysis of such an inter-processor message transfer protocol, called "collect n with timeout". In the next section we derive a suitable queueing model and in Section 3 expressions for the important performance measures are derived. Numerical results are given to illustrate the design options in the choice of the protocol parameters. Protocols with timeouts are difficult to analyse from the performance point of view, and one of the major contributions of this paper is that it does provide an approach to the modelling and exact analysis of protocols with timeout.

2. QUEUEING MODEL

A diagrammatic representation of the queueing model is given in Figure 2. The switch between the SP and PP symbolically represent the virtual connection which is made when messages are to be transferred.

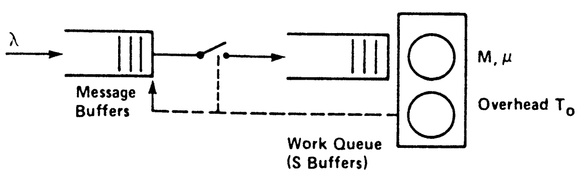

FIGURE 2: QUEUEING MODEL FOR TRANSFER PROTOCOL

We assume the following:

a) Messages arrive at the message buffers (MB) according to a Poisson process with rate λ.

This assumption is based on the fact that messages come from a large number of peripheral processors. The messages are passed to the SP from the periphery under the control of a polling scheme which we do not analyse here. Such work may be found for example in [6].

b) When either n messages have accummulated in the MB or timeout of duration T_m occurs, whichever comes first, the switch closes and messages are transferred to the work queue.

c) Whenever a transfer occurs, overhead time T_o is incurred in the primary processor for the interrupt mechanism.

d) The service time of each message in the PP is assumed to be an i.i.d. random variable with negative exponential distribution function (d.f.) with mean $1/\mu$.

e) The work queue is finite with just S waiting places. Also define $N = S + 1$ (system size).

f) The messages in the MB and work queue are served FIFO.

g) If n more messages accummulate in the MB before the last overhead period of the PP is ended, then the current overhead period is curtailed, the n messages transferred and a new overhead period begun.

This means that the transfer is "gated". This assumption is made for analytical tractability, but in all practical cases seems to be a good one, since the possibility of n messages arriving in a period T_o is remote.

It remains to discuss the system behaviour when the PP work queue becomes full during a message transfer phase. We will assume that when the work queue becomes full, the message transfer is halted, and any messages remaining in the MB must wait until at least the next transfer epoch for transmission to the PP

3. PERFORMANCE ANALYSIS

The analysis in this section drawns upon techniques used in [1,2] and to a lesser extent those in [3,4,6]. Message transfers are initiated at instants $\{t_n\}$. Under the assumptions in Section 2, we observe that the number of messages in the system (MB plus work queue) at the epochs $\{t_n^+\}$ constitutes a Markov chain. The analysis is carried out by utilizing the techniques for the imbedded Markov chain, and by noting that the sequence of inter-transfer intervals constitutes an imbedded semi-Markov process. By choosing the imbedded points just after the transfer epochs, the relationship between inter-transfer interval type and the size of the message batch is preserved. The length of the inter-transfer interval and hence the batch size clearly depends on whether the n messages or the timeout triggers the transfer. Consideration of the inter-transfer interval type is the key to the following analysis.

3.1 State Probabilities

Define the state probabilities P_k of the imbedded Markov chain at the instants $\{t_n^+\}$, and the associated transition probabilities $\{p_{ij}\}$,

$$P_k = \sum_{j=0}^{\infty} p_{jk} P_j \qquad .....(1)$$

To determine the transition probabilities, it is necessary to distinguish among three types of message transfer interval.

Case 1: Timeout. The inter-transfer interval is of length T_m and less than n messages are transferred.

Case 2: n messages in the interval (T_o, T_m), n transferred.

Case 3: n messages in $(0,T_o)$, n transferred.

Define γ_i to be the probability of a transfer interval of type i (i=1,2,3). Conditioning on interval type:

$$p_{jk} = \sum_{i=1}^{3} \gamma_i \, p_{jk}^{(i)} \qquad .....(2)$$

where:

$$\gamma_1 = \overline{E}_n(T_m)$$

$$\gamma_2 = \overline{E}_n(T_o) - \overline{E}_n(T_m)$$

$$\gamma_3 = E_n(T_o) \qquad .....(3)$$

and $E_n(.)$ is the special Erlang d.f. of order n with paramter λ, density $e_n(.)$. In this way it may be seen how the transfer protocol is arrival driven. The three types of interval have probability density functions

(p.d.f.) $f_i(.)$ and d.f. $F_i(.)$ where

$$f_1(t) = \delta(t-T_m)$$

$$f_2(t) = e_n(t)/\gamma_2 \qquad (T_o \leqslant t < T_m) \qquad .....(4)$$

$$f_3(t) = e_n(t)/\gamma_3 \qquad (0 \leqslant t < T_o)$$

The number of service completions between transfers is given by the conditional departure probabilities $d_m^{(i)}$ where

$$d_m^{(i)} = \Pr\{m \text{ service completions} \mid \text{interval type } i\}$$

and are determined as follows:

Case 1: Timeout

$$d_m^{(1)} = (\mu T_s)^m \, e^{-\mu T_s}/m!$$

where $T_s = T_m - T_o$

Case 2: n messages in (T_o, T_m)

$$d_m^{(2)} = \int_o^{T_s} \frac{(\mu x)^m e^{-\mu x}}{m!} f_2(x + T_o)dx =$$

$$\frac{\mu^m \lambda^n e^{-\lambda T_o}}{\gamma_2 m!(n-1)!} \sum_{i=0}^{n-1} \binom{n-1}{i} T_o^{\,n-i-1} \int_o^{T_s} x^{m+i} e^{-(\lambda+\mu)x} dx$$

The type of integral in this equation occurs many times in the subsequent analysis, and a short note on its evaluation is in the appendix.

Case 3: n messages in $(0,T_o)$

$$d_m^{(3)} = \begin{cases} 1 & ;m=0 \\ 0 & ;\text{otherwise} \end{cases}$$

The conditional one-step transition probabilities may now be written down.

Case 2: n messages in (T_o, T_m)

$$p_{jk}^{(2)} = \begin{cases} 0 & ;\ o \leqslant k < n,\ j \geqslant o \\ \sum_{r=j}^{\infty} d_r^{(2)} & ;\ k=n,\ o \leqslant j \leqslant N \\ d_{j+n-k}^{(2)} & ;\ k > n,\ k-n \leqslant j \leqslant k-n+N-1 \\ \sum_{r=N}^{\infty} d_r^{(2)} & ;\ k > n,\ j = k-n+N \end{cases}$$

Case 3: n messages in $(0, T_o)$

$$p_{jk}^{(3)} = \begin{cases} 0 & ;\ o \leqslant k < n,\ j \geqslant o \\ 1 & ;\ k \geqslant n,\ j=k-n \end{cases}$$

Case 1: Timeout

Define $g_i = \Pr\{\text{batch size } i \mid \text{interval of type 1}\}$

$$= (\lambda T_m)^i \, e^{-\lambda T_m}/Ki!$$

where $k = \sum_{j=o}^{n-1} (\lambda T_m)^j \, e^{-\lambda T_m}/j!$

$$p^{(1)}_{jk} = \begin{cases} g_o \sum_{r=j}^{\infty} d_r^{(1)} & ; k = o,\ o \leq j \leq N \\ 0 & ; k = o,\ j > N \\ g_k \sum_{r=j}^{\infty} d_r^{(1)} + \sum_{r=\max(o,j-k)}^{j-1} d_r^{(1)} g_{k+r-j} & ; o < k \leq n-1,\ o \leq j \leq N \\ g_{k+N-j} \sum_{r=N}^{\infty} d_r^{(1)} + \sum_{r=j-k}^{N-1} d_r^{(1)} g_{k+r-j} & ; o < k \leq n-1,\ N \leq j < N+k \\ & \quad k > n-1,\ k+N-n < j < N+k \\ \sum_{r=\max(o,j-k)}^{j-k+n-1} d_r^{(1)} g_{k+r-j} & ; k > n-1,\ k-n+1 \leq j \leq k+N-n \\ g_o \sum_{r=N}^{\infty} d_r^{(1)} & ; k > o,\ j = N+k \end{cases}$$

These transition probabilities are used in equation (1) in conjunction with the power method to obtain a numerical solution to the state probabilities. While the state space is now theoretically infinite, it is truncated to a suitable large value for purposes of numerical computation such that the probability of the largest system state is small (less than 10^{-7}). It is not possible to use a more efficient Gauss-Siedel method because of the existence of transient states.

3.2 Arbitary Time State Probabilities

The arbitary time state probabilities give the distribution of the number of messages in the work queue at an arbitrary observation instant. This distribution is useful in providing a quick calculation of mean delay in the work queue through the use of Little's law. Define $\{P_k^* ;\ o \leq k \leq N\}$ to be the arbitrary time state probabilities. To calculate these it is necessary to know what type of transfer interval is seen by an outside observer at an arbitrary instant. Let π_i be the probability it is of type i. From the properties of the imbedded semi-Markov process we have

$$\pi_i = \gamma_i T_i / \sum_{j=1}^{3} \gamma_j T_j \qquad .....(6)$$

where

$$T_i = \int_o^{\infty} t f_i(t) dt$$

The arbitary time state probabilities require a new set of departure probabilities denoted $\{d_m^{(i)*} ;\ i = 1,2,3\}$

Case 1: Timeout. Given that the observer is in type 1 interval, he is in the overhead segment with probability π_{11} and the "active" service (non-overhead) segment with the complementary probability π_{12} where

$$\pi_{11} = T_o/T_m = 1 - \pi_{12}$$

The backward recurrence time for the active service segment has p.d.f.

$$\overline{F}_1(x)/T_s = 1/T_s \qquad (o \leq x < T_s)$$

so that

$$d_m^{(1)*} = \frac{1}{T_s} \int_o^{T_s} \frac{(\mu x)^m e^{-\mu x}}{m!} dx$$

$$= \frac{\mu^m}{T_s m!} \int_o^{T_s} x^m e^{-\mu x} dx$$

Case 2: Given that tha observer is in a type 2 interval, he is in the overhead segment with probability π_{21} and the active service segment with the complementary probability π_{22} where

$$\pi_{21} = T_o/T_2 = 1 - \pi_{22}$$

Denote by $T_2' = T_2 - T_0$ the mean length of the active service segment of a type 2 interval

$$T_2' = \int_o^{T_s} x f_2(x + T_o)\, dx =$$

$$\frac{\lambda^n e^{-\lambda T_o}}{\gamma_2 (n-1)!} \sum_{i=o}^{n-1} \binom{n-1}{i} T_o^{n-i-1} \int_o^{T_s} x^{i+1} e^{-\lambda x} \, dx$$

Finally

$$d_m^{(2)*} = (1/T_2') \int_o^{T_s} \frac{(\mu x)^m e^{-\mu x}}{m!} \overline{F}_2(x+T_o)\, dx$$

which can be reduced by substitution.

Case 3: Here there is no possibility for departures.

The distribution of the number of messages in the work queue at an imbedded instant is denoted by $\{P_k'\}$ where

$$P_k' = \begin{cases} P_k & ; o \leq k < N \\ \sum_{r=N}^{\infty} P_r' & ; k = N \end{cases}$$

Finally,

$$P_k^* = \pi_1(\pi_{11} P_k' + \pi_{12} \sum_{j=k}^{N} d_{j-k}^{(1)*} P_j') \qquad (o < k \leq N)$$

$$+ \pi_2(\pi_{21} P_k' + \pi_{22} \sum_{j=k}^{N} d_{j-k}^{(2)*} P_j')$$

$$+ \pi_3 P_k' \qquad .....(7)$$

$$P_o^* = \pi_1(\pi_{11} P_o' + \pi_{12} \sum_{j=0}^{N} P_j' \sum_{r=j}^{\infty} d_r^{(1)*}) \quad (k=0)$$

$$+ \pi_2(\pi_{21} P_o' + \pi_{22} \sum_{j=o}^{N} P_j' \sum_{r=j}^{\infty} d_r^{(2)*})$$

$$+ \pi_3 P_o'$$

3.3 Mean Waiting Time in Work Queue

Using in principle the methods in [1], the expected delay in the work queue W_q is found from Little's Law [7] as

$$W_q = W - 1/\mu$$

where $W = L/\lambda$ and $L = \sum_{k=o}^{N} kP_k^*$

3.4 Transfer Delay

The mean transfer delay is made up of two components, the mean delay from message arrival until the next transfer epoch denoted by the expectation W_{t1} and the mean delay from the first transfer epoch until that transfer epoch the message is put into the work queue, denoted by the expectation W_{t2}. This decomposition is possible because except for fresh arrivals, the number of messages in the secondary buffers remains constant between transfer epochs. From Little's Law,

$$W_{t2} = \sum_{k=N+1}^{\infty} (k-N) P_k/\lambda$$

The other component depends on interval type. Using the results in [5], the probability an arbitrary message coming in a certain type interval is related to the expected number of message arrivals in that interval. Define S_i to be the expected batch size for interval type i

$$S_i = \begin{cases} \sum_{k=o}^{n-1} k\, g_k & ; i = 1 \\ n & ; i = 2,3 \end{cases} \qquad .....(9)$$

Therefore

a_i = Pr {arbitrary message arrives in interval type i}

$$= \gamma_i S_i / \sum_{j=1}^{3} \gamma_j S_j \qquad .....(10)$$

Given there is no blocking of messages, we have

$$W_{t1} = \sum_{i=1}^{3} a_i W_{t1}^{(i)}$$

where

$$W_{t1}^{(i)} = \begin{cases} T_m/2 & ; \; i=1 \\ (n-1)T_i/2n & ; \; i=2,3 \end{cases}$$

Finally, the total mean transfer delay, W_t, is given as

$$W_t = W_{t1} + W_{t2} \qquad \ldots\ldots(11)$$

3.5 System Stability

Similar to [2], there are two conditions deciding the system stability. Firstly, the expected number of arrivals in an arbitrary inter-transfer interval may not exceed the expected number of service completions. Secondly, the expected number of arrivals may not exceed the PP system size. If either condition is violated, the number of messages in the whole system grows without bound.

Condition 1:

$$\gamma_1 \lambda T_m + n(\gamma_2 + \gamma_3) < \gamma_1 \mu T_s + \gamma_2 \mu T_2' \qquad \ldots(12)$$

Condition 2:

$$\gamma_1 \lambda T_m + n(\gamma_2 + \gamma_3) < S + 1 \qquad \ldots(13)$$

These conditions describe functional equations for the protocol parameters (n, T_m) which must be obtained iteractively. For a given value of n, eqns. (12) and (13) define respectively minimum and maximum stable values of T_m, denoted T_{min} and T_{max}. If n becomes very large, the limits derived in [2] apply, namely $T_{min} = T_o/(1-\rho)$ where $\rho = \lambda/\mu$ and $T_{max} = (S+1)/\lambda$. Also note for $n < S$, T_{max} goes to infinity. In practice, this will usually be the case. Results from T_{min} and T_{max} are depicted in Figure 3.

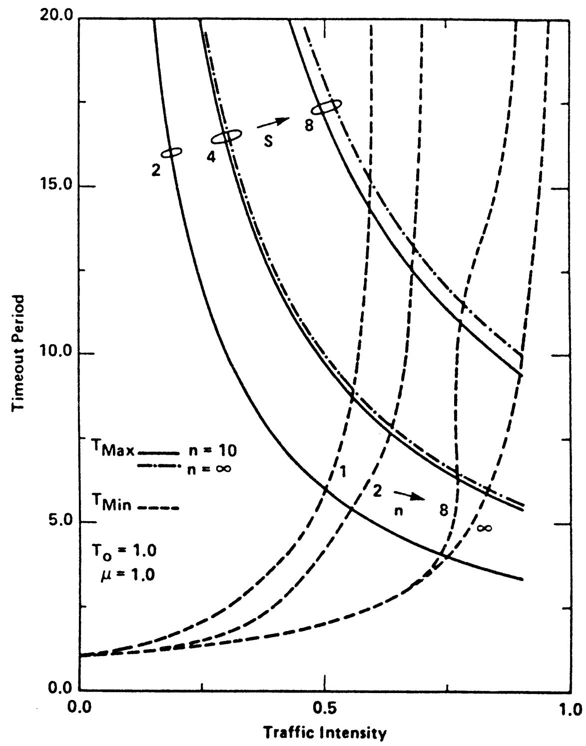

FIGURE 3: SYSTEM STABILITY LIMITS

4. RESULTS AND DISCUSSION

The main results of the analysis concern the total mean messaging delay, including processing delays in the work queue. This is a function of the protocol parameters (n, T_m), as well as of the underlying level of message traffic. Figure 4 depicts total mean delay as a function of offered traffic intensity. The most noticable effect is for $T_m = 10$, $n = 4$ where the "batching" of messages for transfer can be seen to lower the total delay with increasing traffic. This occurs because the probability of n messages arriving before the timeout expires increases with increasing traffic load. Eventually, of course, this becomes self defeating because at very high load, transfers become to be initiated too frequently and too much time is lost to overhead.

Figure 5 depicts total mean delay versus the timeout period. The optimising effect of the timeout is here apparent. For T_m small, too much time is lost in overhead, and for T_m large, the delay becomes limited by the message batch size and traffic level, i.e. how long does it take to collect the n messagesfor transfer.

5. CONCLUSIONS

We have provided here an exact analysis of a queueing model for a message transfer protocol between processors in a distributed system. The results show how the protocol parameters may be chosen to minimise messaging delays, and be robust with respect to the amount of messaging traffic. Minimisation of messaging response times is very important in distributed SPC systems for maintaining call completion rates at high load. The analysis presented here has further applications for other types of real-time systems, for example bus systems and packet-switching systems.

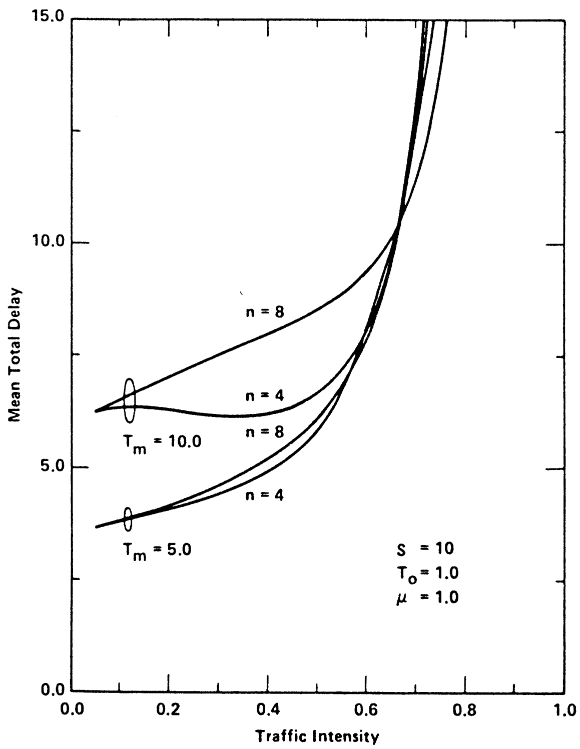

FIGURE 4: TOTAL MEAN DELAY vs OFFERED TRAFFIC

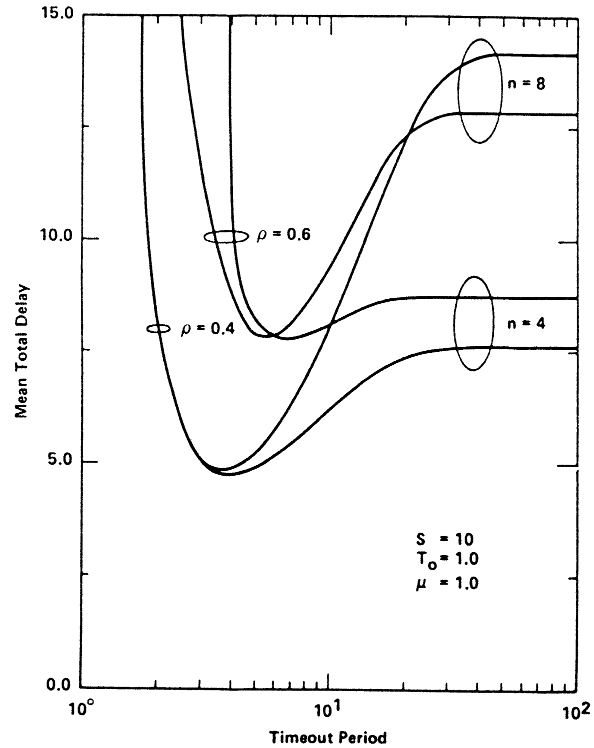

FIGURE 5: TOTAL MEAN DELAY vs TIMEOUT PERIOD

ACKNOWLEDGEMENT

The authors would like to thank H. Van As for help with the simulations which were used for validating the results.

Appendix - Evaluation of Partial Integrals

In many places integrals of the following form are generated

$$I = \int_o^T x^n e^{-\lambda x} dx$$

$$= (n!/\lambda^{n+1})(1 - \sum_{k=o}^{n} ((\lambda T)^k e^{-\lambda T})/k!)$$

If λT is small, it may be better to use instead the following form

$$I = (n!/\lambda^{n+1}) \sum_{k=n+1}^{r} ((\lambda T)^k e^{-\lambda T})/k!$$

with r chosen sufficiently large.

REFERENCES

[1] D.R. Manfield and P. Tran-Gia, "Queueing Analysis of Scheduled Communications Phases in Distributed Processing Systems", Proc. 8th Int. Symp. on Computer Performance Modelling, Measurement and Evaluation, Nov. 1981, Amsterdam.

[2] P. Tran-Gia and H. Jans, "Delay Analysis of Clocked Event Transfer in Distributed Processing Systems", Proc. 6th ICCC, London, 1982.

[3] M. Langenbach-Belz, "Two-Stage Queueing System with Sampled Parallel Input Queues", Proc. 7th I.T.C., Stockholm, 1973.

[4] B. Powell and I. Ivi Itzhak, "Queueing Systems with Enforced Idle Times" Oper. Res., 15, 1145-1156 (1967).

[5] P.J. Burke, "Delays in Single Server Queues with Batch Input", B.S.T.J., 23 830-833 (1975).

[6] P.J. Kuehn, "Analysis of Switching System Control Structure by Decomposition", Proc. 9th I.T.C., Spain, 1979.

[7] R.B. Cooper, "Introduction to Queueing Theory", Macmillan, 1972.