Evaluation tools for real-time message scheduling algorithms: Analytic solver versus event-driven simulator

Philippe Jacquet, S. Sedillot

INRIA

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

N° 638

EVALUATION TOOLS FOR REAL-TIME MESSAGE SCHEDULING ALGORITHMS: ANALYTIC SOLVER VERSUS EVENT-DRIVEN SIMULATOR

Philippe JACQUET
Simone SEDILLOT

Mars 1987

EVALUATION TOOLS FOR REAL-TIME MESSAGE SCHEDULING ALGORITHMS :

Analytic solver versus event-driven simulator

OUTILS D'EVALUATION D'ALGORITHMES D'ORDONNANCEMENT DE MESSAGES EN TEMPS REEL :

Modèle analytique et simulateur évènementiel

Philippe. JACQUET Simone. SEDILLOT

INRIA - Projet SCORE
BP 105 78153 LE CHESNAY CEDEX FRANCE

Abstract : Objectives are to have tools for real-time local area networks equipments performance evaluation. ISO OSI-RM layered architecture is modelled into a queueing model. Both a delivery deadline and a static priority are specified for each message that is transmitted. A local time dependant scheduler is defined whose execution time upper bound can be predicted. An event-driven simulator is presented as well as an analytic model ; Experiments are reported that proove that both running under the same assumptions provide same results. Because the analytic model does not include promptness control, experiments are reported that determine conditions on traffic load and timing constraints as well as on layer service partitionning into tasks under which the analytic model either cannot be used, or, reversely, can be advantageous in terms of runtime.

Résumé : Nous avons comme objectif la réalisation d'outils pour évaluer les performances d'équipement pour réseaux locaux à contraintes temps réel. L'architecture en couche ISO OSI-RM est modèlisée par un système de files d'attente. Pour chaque message à transmettre on spécifie une échéance et une priorité statique. Ces quantités sont gérées par un ordonnanceur temps réel dont on peut évaluer les coûts maximaux d'exécution. Nous présentons un simulateur évènementiel ainsi qu'un modèle analytique simple. La cohérence entre les deux est montrée par le fait qu'ils produisent les mêmes résultats lorsqu'ils évaluent les mêmes systèmes. Néanmoins le modèle analytique peut s'avèrer deficient lorsque les conditions de trafic sont telles qu'elles induisent de forts taux de rejet (le modèle analytique n'inclut pas un contrôle de promptitude effectif) ou lorsque les algorithmes de fractionnement des taches des services deviennent difficile à modèliser. Cependant, en condition normale (faible taux de rejet, archtitecture simple), le modèle analytique est avantageux en terme de temps de calcul.

EVALUATION TOOLS FOR REAL-TIME MESSAGE

SCHEDULING ALGORITHMS :

Analytic solver versus event-driven simulator

Philippe. JACQUET Simone: SEDILLOT
INRIA - Projet SCORE
BP 105 78153 LE CHESNAY CEDEX FRANCE

I. INTRODUCTION

The work related in this paper is part of a performance evaluation platform for real-time local area networks. Real-time should be understood as related to process control, avionics, flexible plants environments. Stations connected on the network are supposed to be designed according to the OSI ISO-RM layered architecture I ISO 81 I. Real-time quality in servicing is expressed by a delivery deadline and static priority assigned to each message by the application. Local executive is in charge of enssuring that messages are effectively processed in time to meet their delivery time. In case of CPU conflict, priority is given to messages with lower static priority values (higher priorities). A scheduling algorithm is proposed whose major property is to have an upper-bounded execution time in the order of 150 instructions.

An event driven simulator is developped to evaluate the station architecture performances. The simulation is able to produce results concerning scheduling overheads, tasks set up times, CPU utilization and message processing time distributions.

Late messages are rejected, I LEL 85 I that is, signalled as error to the application. Hence the main performance critera is rejection rate.

The simulator is flexible and incremental in terms of sophistication that is, it can support any additive software for simulating communication entities. But presently, each entity is modelled by a constant layer service time. Entity-task mapping is a parameter.

Because of obvious run time limitation, an analytic model is developped for performance evaluation at light load. The analytic model is used to validate the simulation.

The scheduler design is presented in chapter II. Chapter III, IV an V concern the simulator outline, performance metrics and the analytic model design. Chapter VI relates experiments that have already been performed,namely :

- Validation of the simulator,
- The setting of loads and layer-tasks mapping that cannot be accounted for by the one file analytic solver,
- Proof of the scheduler efficiency at any load between 0 and 80 %.

II. QUEUEING MODEL FOR AN ISO OSI-RM LAYERED ARCHITECTURE MODEL WITH TIMING CONSTRAINTS ON MESSAGE DELIVERY TIME (Figure 1)

Each layer i is modelled by a waiting queue SQ(i) and a maximum layer service time S(i). It is possible to introduce random service jittles. Any number l of layers can be modelled.

A message transmission is instantiated by a SendMessage request.

SendMessage requests enter the model in arrival queue AQ. The lowest layer that can be modelled is layer 2 restricted to LLC functionnalities. MAC layer is not included since processing at MAC level is driven by the physical medium state [IEE 84, IEE 85] and, thus, is not relevant to purely local scheduling.

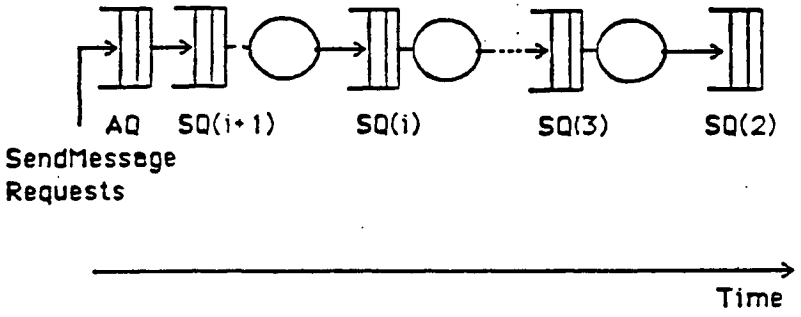


Figure 1 : OSI-RM layered architecture model

II.1. Message deadlines

For any SendMessage (m) request, let T(m,2), T(m,3) ..., T(m, i) be m delivery deadlines at layer 2, 3 ..., i respectively. We will assume that T(m,2) is specified by the user. If m is at layer i, we call T(m,i) its current deadline.
Defining $R_p(i)$ as layer i response time -including waiting time in SQ(i) and service time S(i) - and assuming that $R_P(i)$ can be predicted, m deadline at layer i + 1 is

$$T(m,i+1) = T(m,i) - Rp(i+1)$$

II.2. Message static priority

The user is allowed to specify a static priority j(m) for each request.

II.3. Message time-dependent global priority

Assuming that current time is t and a request is at layer i, its global priority is :

$$H(m,i,t) = j(m) + a \cdot (T(m,i)-t).$$

If weight parameter "a" is a constant, the difference between any two messages global priority values is a constant, whatever is t. Thus, H(m,i,t) can be made

$$H(m,i) = j(m) + a \cdot T(m,i).$$

If m is at layer i, we define H(m,i) as m current global priority. Special messages called alarms, have all their deadlines set to zero which give them lowest global priorities provided their static priority is always smaller than that of non alarm (regular) messages.

II.4. Message scheduling

So that the scheduler execution time might be upper bounded, we define scanning windows in each queue. Let W be the window size.
The algorithm we use guarantees that, at any time it is activated, the scheduler processes the W requests with the smallest global priority values out of the l • W requests with shortest current deadlines.

The scheduler functions are the following :

Step 1 : For each of the first W requests - if any - in arrival queue AQ, and for each layer i, calculate T(m,i) and H(m,i), then put these requests in SQ(1). Each SQ(i) is ordered by increasing values of current deadlines.

Step 2 : Sort the W requests ahead of each SQ(i)s in the increasing order of current global priority values

Step 3 : Create a time scale with the W first requests of the sorted list.

II.5. Message processing

A Real-Time Monitor, MRT, processes the time scale in the FIFO order. It allocates the CPU to the layer entity awaited by the processed request.
Scheduling occurences can be either event driven -events being end of service or request arrival times- , periodic or pseudo-periodic with a jittle equal to the MTR maximum non-preemption delay.
Only one CPU is considered, nevertheless the scheduler could create as many time scales as CPUs if a multiprocessor is used.

II.6. Promptness control

At any time t, promptness control can be executed on a SendMessage request at layer i : Namely, if T(m,i) > t, the request is rejected of the system. Obviously, layer response times tuning affects rejection rates : they can make rejection either occur too early and thus lead to unnecessary rejections, or too late and missuse the CPU resource.

Presently, promptness control is performed by the scheduler

- on all requests extracted from the arrival queue,
- on all SQ(i)s,

and by the MTR

- immediatly after each request service time.

It can be inhibited if desired for specific runs.
Alarm messages are never submitted to promptness control

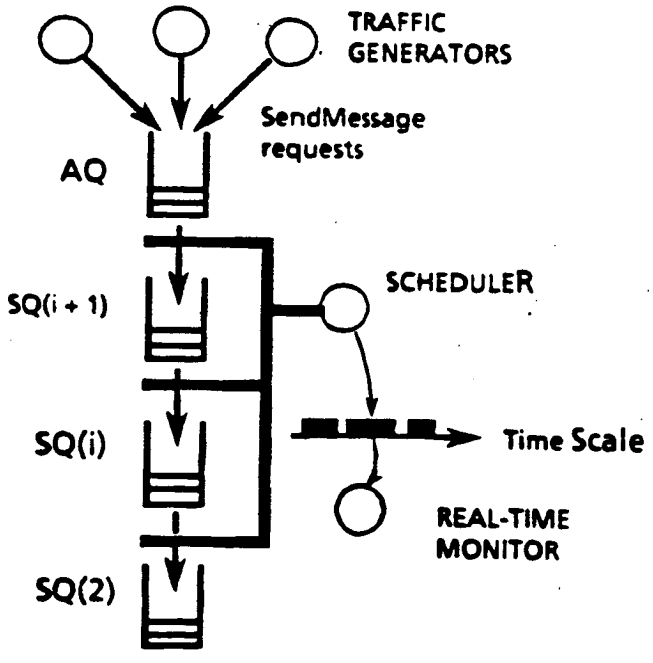


Figure 2 : The simulator model

III. THE SIMULATOR MODEL (Figure 2)

The model in the simulator entails any number of layers, each layer being modelled by a queue and a service time.

The scheduler is completly implemented. Its execution time is accounted for according to the number of instructions it executes.
The MRT simulates CPU allocation by increasing the simulated time by the requested layer service time.

Sampling are implemented to measure request execution times, rejection rates and layer response times. Traffic generators can produce requests with either periodical, poisson or avalanche distributions. Static priority and deadline at layer 2 are specified for each generator.

IV. PERFORMANCE METRICS

Scheduling scheme performance is measured by both message rejection rates per traffic and mean, maximum and standard deviation of alarm execution times. Nevertheless for the sake of more detailed analysis, simulator runs provide request execution time distributions per traffic, queues client number distributions and layer response times.

V. THE ANALYTIC APPROACH

There is an approximate modelization which deals with local arranging within dynamical priorities. This system can be exactly analyzed without expensive computation. .

V.1. The scheme

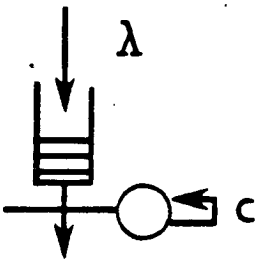


Figure 3 : Single file scheme

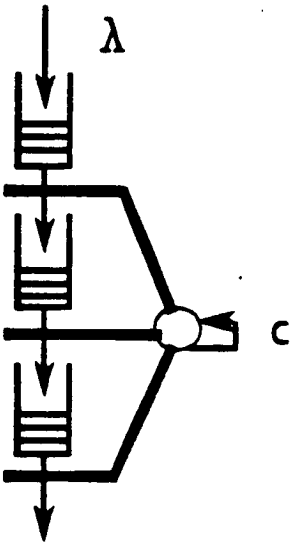


Figure 4 : Multi-file scheme

V.1.1. - The periodic server

We consider a queue which is served by a periodic server : at every begining of a period the server comes, takes the lower priority holder -if any- leaves the queue and comes back at the beginning of the following period.

Let c be the constant length of the period.

The advantage of this scheme consists in the fact that the server behaviour does not interfer with the generated state of the queue. The main approximation with the non periodic - obvious

scheme (continuous), with deterministic service time c, is in the idle duration of the server. In the periodic scheme, idle periods of time are integer multiples of the period unit, in the continuous scheme they can be any arbitrarily intermediate real values.

At the end of this section we evaluate the multi-files scheme in which m successive queues are served by a simple periodic server of period c.

V.1.2. Aperiodic traffic

Message arrival distribution is of Poisson type with rate λ. It can be the superposition of Poissonian subtraffics of respective rates $\lambda_1, \lambda_2, \lambda_3 \ldots (\Sigma \lambda_i ; = \lambda)$. Every subtraffic corresponds to a deterministic type of messages (deterministic priority).

V.1.3. Dynamical priorities

A new message holds an initial priority T corresponding to its type. This priority is of the form

j + delay.

This scalar is decreased like a delay as long the message stays in the queue (continuous decreasing of one unit per time unit).

V.1.4. No promptness control

This allows priorities on the negative axis. As far the user is concerned, we hope that the rejection rate is very low. The "effective" promptness control will introduce only a minimal perturbation (in the proportional range of this rate of rejection itself) in the simulator. Thus we can approximate the "effective" rejection rate due to promptness control by a "virtual" rejection rate computed with the probability of overpassing one's deadline (but without being

rejected). This evaluation will be accurate in the proportional range of this probability itself. For example, if we find 1% of virtual rejection rate, we can evaluate to 1% ± 0.01% of effective rejection rate. The evaluation can be more restricted in the range [1% - 0.01%, 1%], because it is quite obvious that the "virtual" rate is higher than the "effective" one, (serving obsolete messages delays in the non obsolete ones).

V.2. Form of the results

We evaluate the spectrum of waiting time for every type of messages (*i.e.* for every initial priority T_1, T_2, T_3...

V.3. Analysis of the single queue scheme

V.3.1. State of the queue for a priority T

Let T be a real number. At the begining of an arbitrary period we consider the number N of customers (messages) in queue with priority less than T. From the begining of the period to its end, we have the probabilistic transition :

nb of customers of priority ≤ T		nb of customers of priority ≦ T-c
N + 1	→	N + X(a(T))
0	→	X(a(T))

where a(T) is the *poissonian* rate of generation on a period of new customers which will have a priority less than T-c at the end of the period. X(a) is the random variable associated to the number of arrival when the distribution is poissonian of rate a. (V.1.1)

$$Proba(X(a) = k) = \frac{a^k}{k!} e^{-a}$$

Let $q_{n,T}$ be the stationary probability for having N = n. Let q_T be the generating function of the stationary distribution of N.

$$q_T(z) = \sum q_{n,T} \cdot z^n$$

where z is a complex variable.

According to (V.I) we have :

$$q_{T-c}(z) = \left[\frac{q_T(z) - q_T(0)}{z} + q_T(0) \right] e^{-a(T).(1-z)} \tag{V.2}$$

Let Tmax the highest initial priority of the traffic. We resolve the previous equation with $T = T\text{-}c = \infty$.

$$q_{T_{max}}(z) = (1-\lambda).\frac{1-z}{1-ze^{\lambda(1-z)}} \tag{V.3}$$

Finally, we resolve (V.2) with a downward recursion with starting point (3).

V.3.2. Waiting time for the priority T

Let $K_{n,T}$ be the number of waiting periods for a customer of priority T at the begining of the first period with already n other customers in queue of lower priority. Let $A_{n,T}{}^K$ be the stationary probability for having K = k.
From the first period to the second we have the transition :

first period		second period
$AK_{n+1,T}$	=	Σ Proba(Xa(T)) = X).$A_n{}^{K-1}{}_{+X,T-c}$
$A0_{0,T}$	=	1 (the customer is immediatly served)

(V.4)

Let $C_{n,T}$ be the characteristic function of the distribution of $K_{n,T}$.

$$C_{n,T}(u) = \sum A^k_{n,T} u^k$$

Let C be the generating function related to n of the previous expression :

$$C_T(z,u) = \sum_n C_{n,T}(u) z^n$$

Let "*" be the operand between generating functions f and g

$$f(z) = a_0 + a_1 z + .. + a_n z^n + ...$$

$$g(z) = b_0 + b_1 z + .. + b_n z^n + ...$$

$$f(z) * g(z) = \sum_n \left[\sum_{l-K=n} a_l b_K \right] z^n$$

or

$$b(\frac{1}{z}) * a(\frac{1}{z}) + a(z) * b(z) - \sum_n a_n . b_n = a(z) b(\frac{1}{z}).$$

According to (V.4) we have :

$$C_T(z,u) = 1 + uz.C_{T-c}(z,u) * e^{-a(T).(1-z)} \tag{V.5}$$

Let T_{min} be the lowest initial priority of the traffic. There is not any new customer which can precede in queue a customer which has already reached a priority less than T_{min}. Thus :

$$K_{n,T_{min}} = n \ ;$$

or

$$C_{T_{min}}(z,u) = \frac{1}{1-uz} \tag{V.6}$$

Finally we resolve (V.5) with a upward recursion with starting point (V.6).

V.3.3. State of the results

Let K_T be the number of waiting periods for a customer with priority T at the begining of the first period and W_T its characteristic function. Of course it holds :

$$W_T(u) = \sum_n q_{n,T} . C_{n,T}(u). \tag{V.7}$$

Let w_i be the characteristic function of the number of waiting periods for a customer of initial priority T, we get :

$$w_i(u) = \int_0^c C_{T_i - r}(u) dr \tag{V.8}$$

Let $w^*_i(s)$ the Laplace transform of the toal waiting time of a message of type i (included the fraction r of generation period and the duration f of the effective service in a period).

$$w_i^*(s) = \int_0^\infty Proba\ (waiting\ time \in \left[t, t+dt\right]) e^{-ts} ;$$

$$w_i^*(s) = \int_0^c e^{-s\,r} C_{T_i - r}(e^{-s\,c}) e^{-fs} \tag{V.9}$$

V.4. Analysis of m files in succession with a single periodic server

There are two ways for dealing with the priorities.

(i) The final delay priority : in this scheme the delay included in the expression of the priority is the delay from the current time to the expected exit of the last file. There is no modification of the priority between two files, except the continous devaluation.

(ii) The intermediate delay priority : in this scheme the delay included in the expression of the priority is the delay from the current time to the expected exit of the current file. Thus there is a modification of the priority between two files which corresponds to the addition of c to the previous delay in order to evaluate the new one.

In the multi-files scheme a new message of priority T corresponds to m new customers in a single file with respective priorities all equal to T if we are in final delays or distributed as T, T + c,...,T + (n-1)c if we are in intermediate delays.

V.4.1. The final delay priority

There are slight modifications of the reasonning from the analysis of the single queue scheme. At first (V.2) and (V.3) becomes :

$$q_{T-c}(z) = \left[\frac{q_T(z) - q_T(0)}{z} + q_T(0) \right] e^{-a(T)(1-z^m)} \tag{V.2.1}$$

and

$$q_{T_{max}}(z) = (1-m\lambda).\frac{1-z}{1-ze^{\lambda(1-z^m)}}.$$

(V.3.1)

A same modification occurs in (V.5) :

$$C_T(z,u) = 1 + uz.C_{T-c}(z,u)^* e^{-\alpha(T).(1-z^m)}$$

(V.5.1)

The m pieces of a message hold the same priority T but we assume that there is a certain ordering between them. The waiting time of the last one is the waiting time of the full message. Now (V.7) is :

$$W_T(u) = \sum_n q_{n+m-1,T}.C_{n,T}(u).$$

(V.7.1)

V.4.2. The intermediate delay priority

The differences from the analysis of the final delay priority are in new expressions of (V.2) and (V.5) :

$$q_{T-c}(z) = \left[\frac{q_T(z)-q_T(0)}{z} + q_T(0)\right] e^{-\sum_{l=1}^{m}\{\alpha(T-(l-1)c)-\alpha(T-lc)\}.(1-z^l)}$$

(V.2.2)

and

$$C_T(z,u) = 1 + uz.C_{T-c}(z,u)^* e^{-\sum_{i=1}^{m}\{\alpha(T-(l-1)c)-\alpha(T-lc)\}.(1-z^l)}$$

(V.5.2)

Of course we must care that T_{max} is now T_{max} + (m-1)c in (3).

VI. EXPERIMENTS

VI.1. Objectives and parameters

Experiments, objectives are threefolds :

- First, validate the simulator by comparison of its results with the analytic solvers's,
- Secund, identify traffic characteristics, scheduling and service-task mapping options for which the one file analytic model cannot be used.
- Third, consolidate our intuitions on scheduling efficiency.

. Layers service time

Total service time per request, S, is set to one thousand Pascal instructions execution time. Both, experiments in protocol implementation and actual MAP trends [MAP 85] motivates this choice.

. Scheduler maximum exec ution time assumptions

It is out of scope in this paper to detail the process, but the scheduler maximum execution time max $C_\varnothing$, expressed in terms of instructions number, can be estimated given its specifications (see above II.4). The only undeterministic element in the scheduler execution time upper bound is the maximun mumber of requests in queues denoted max Q. Assuming IKNU 73I works, and without going in the details of calculations, it comes for the scheduler execution time upper bound.

$$\max C_\varnothing = W^*\{N_i^*(\alpha + \beta \log_2 W^* N_i) + \beta \log_2 \max Q + 3\gamma\}$$

with

N_i = number of layers

α = T(m,i) and H(m,i) calculation time,

β = criteria comparaison time,

γ = entering of a request at a specified rank in a queue.

Numerical applications and measurements of max $C_\varnothing$ through simulations are reported at the end of the paper (Table 5 and Figure 10) but it is of major interest at this point to note that measurements validate totally the a priori calculation alove. This is fundamental, should the scheduling be garanteed to occur periodically.

Our estimated max $C_\varnothing$ values range from 40 to 200 instructions when the number of layer is smaller or equal to 3 and a window value smaller or equal to 3. Given these estimations, we decided to consider, in our experiments, the following values for max $C_\varnothing$: S/8, S/6 and S/4.

VI.2. Experiments 1 : Simulator validation

The one file analytic solver and the simulator performances are compared when working under the same assumptions, namely :

- one layer is modelled in the simulator,
- promptness controls are inhibited in the simulator since there are none in the analytic model.
- scheduling occurs periodically in the simulator just as service is periodic in the analytic model.

Setting the scheduling period $P_\varnothing$ in the simulator as well as the service time P_{AM} in the analytic model, depends on max $C_\varnothing$, W and layer service time values, according to the relation :

$$P_{AM} = P_\varnothing = \max C_\varnothing * W * S$$

In this step, W is set to one since such is the case in the analytic model, by definition. Thus, period values are either equal to S * (1 + 1/8), S * (1 + 1/6) or S * (1 + 1/4).

- Traffics :

Two sets of generators, respectively named G_{38} and G_{78} after the CPU load they represent, are used. Each set consists of three generators g1, g2 and g3. Generators generate requests according to a Poisson distribution law. G_{38} and G_{78} characteristics are given on tables 1. Instruction execution time is made equal to two microseconds -which is the announced time for M68020 microprocessor-. Thus, g1, g2 and g3 request maximum allowed execution delays and interarrival times are expressed in terms of instruction unit. g(1) and g(2) have the same mean interarrival time and maximum allowed execution delay ; The purpose of this is to observe the inpact of their relative static priority. Note that service time and allowed execution delays expressed in time unit range from 5 to 20 ms which is the order of magnitude announced by NBS [MIL 86] and experts in the domain of launching vehicules [DUR 85].

Generator ident	g1	g2	g3
λ(g)	5 * S	12 * S	12 * S
D(g)	2.5 * S	10 * S	10 * S

Table 1.1 : G_{38} characteristics

Generator ident	g1	g2	g3
λ(g)	2.5 * S	6 * S	6 * S
D(g)	2.5 * S	10 * S	10 * S

Table 1.2 : G78 characteristics

λ(g) = Resquest mean interarrival time for traffic g
D(g) = Resquest maximum allowed delay for traffic g

Table 1 : Traffic characteristics

Comparison of the results :

Traffic static priority		0	1	2	2	1	0	
Rejection rates		R1	R2	R3	R1	R2	R3	
$P_\emptyset = P_{AM}$	$\lambda/P_\emptyset$							
1.125 * S	4.933	8 9	0 0	0 0	8 10	0 0	0 0	Solver Sim
1.166 * S	4.759	9 11	0 .4	0 .5	8 12	0 0	0 0	Solver Sim
1.250 * S	4.440	11 10	.4 .5	.5 .5	11 14	.4 .5	.4 .4	Solver Sim

Table 2.1 : Rejection rates for G_{38}

Traffic static priority		0	1	2	2	1	0	
Rejection rates		R1	R2	R3	R1	R2	R3	
$P_\emptyset = P_{AM}$	$\lambda/P_\emptyset$							
1.125 * S	2.462	38 32	28 20	25 22	37 39	22 28	18 25	Solver Sim
1.166 * S	2.375	45 40	28 28	31 30	43 47	30 31	26 28	Solver Sim
1.250 * S	2.216	67 60	56 48	56 48	67 65	56 50	53 48	Solver Sim

Table 2.2 : Rejection rates for G78

Table 2 : Simulator validation : Comparison of rejection rates produced by both the analytical solver and the simulator.

Rejection rates R1, R2 and R3, related to traffic generators g1, g2 and g3 respectively, are reported on tables 2.1 and 2.2. The maximum deviation is 7 units with traffics G_{78}. Since solver and simulator runs on different machines, this deviation is explained by processing sensitivity to minor differences in poisson distribution mechanisms, given the small value of the ratio

$$\frac{\textit{Global mean request interarrival time}}{\textit{request minimum execution time}} .$$

Indeed, global mean request interarrival time λ is such that

$$\frac{1}{\lambda} = \Sigma \frac{1}{\lambda(g)} \qquad \textit{for } g = 1, 2, 3.$$

and minimum execution time is $P_\emptyset$. Numerical evaluations show that $\lambda / P_\emptyset$ values range from 2.216 to 4.933.

VI.3. Experiments 2 : Analytic solver validity domain

In the following experiments, the solver parameters are unchanged (see VI.2). The simulator is set to model three layers ; Hence layers service time S per request is splitted into three service times of S/3 duration for each layer ; Such a straigthforward option is motivated by the actual lack of precision concerning forthcoming real-time protocols. The setting of periods values depends on W since

$$P_{\emptyset} = \max C_{\emptyset} + W \cdot S/3$$

The choice of W is dictated by the fact that request mean execution time becomes 3 times $P_{\emptyset}$. Given g1 maximum allowed delay of 2.5 • S, W is maintained equal to 1 which provides the minimum values for the ratio

$$R = \frac{maximum\ allowed\ execution\ delay}{minimum\ execution\ time}$$

namely 1.429 to 1.818 for max $C_{\emptyset}$ ranging from S/8 to S/4.

Promptness controls are set in the simulator.

Rejection rates per traffic are drawn on Figures 5.1 and 5.2, relatively to those produced by the solver in experiment 1. Rejection rates produced by the simulator are always higher than the solver's and deviations are higher with G_{38} than with G_{78}. Explanations of this are deduced when analyzing, on one hand, CPU utilization percentages (Figures 6.1 and 6.2) and, on the other hand, layer service capacity and request minimum execution time variations (Table 3) in both theanalytic model and the simulator : Layer service capacity for both the analytical model and the simulator is defined by the current value of the ratio

$$C_a(s) = \frac{maximum\ layer\ service\ time\ por\ period}{period\ time}$$

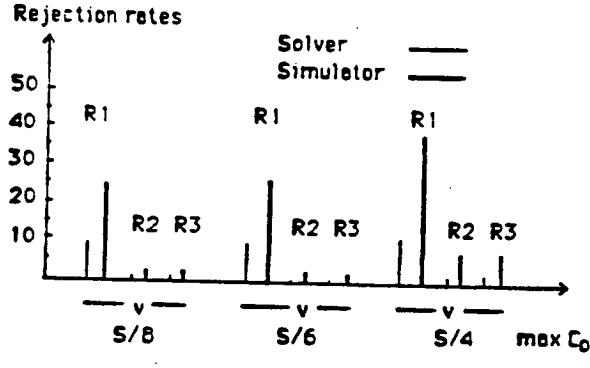


Figure 5.1 : Rejection rates with G_{38}

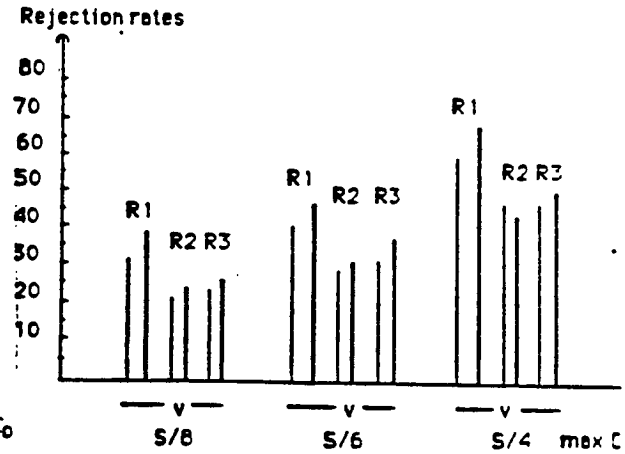


Figure 5.2 :Rejection rates with G_{78}

Figure 5 :Analitic model validity domain. rejection rates comparison

. With G_{38}, in both analytic model and simulator, it appears that CPU utilization for layer service (Figure 6.1) is always used below the capacity and equal to the load. Consequently, we deduce that all requests are fully served and rejections occur at request last service termination time. Then we can legitimely state that rejection rate is sensitive to the request minimum execution time variations only and not to layer service capacity variations. Sensitivity is shown on Figure 7.

One may argue that when R does not exceed the numerical value of 2, setting the analytic model service period to

$$P_{AM} = S + \max C_{\emptyset}$$

does not reflect properly the behavior of the simulator for which the period is scheduling

$$P_{\emptyset} = S/3 + \max C_{\emptyset}$$

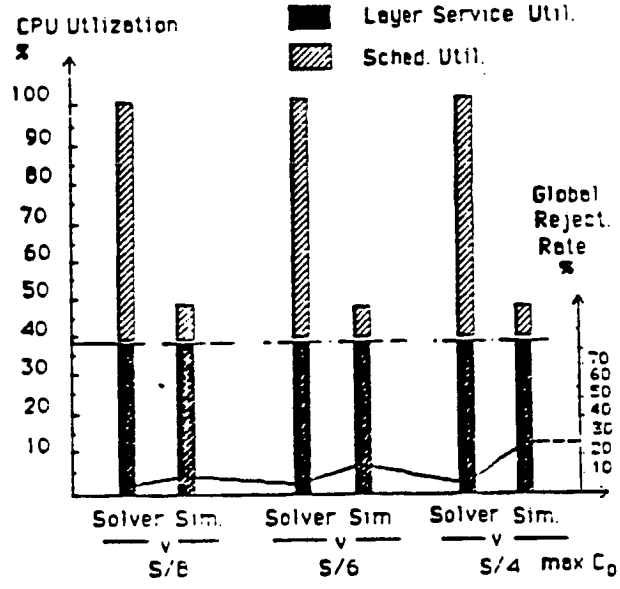


Figure 6.1: CPU Utilization with G_{38}

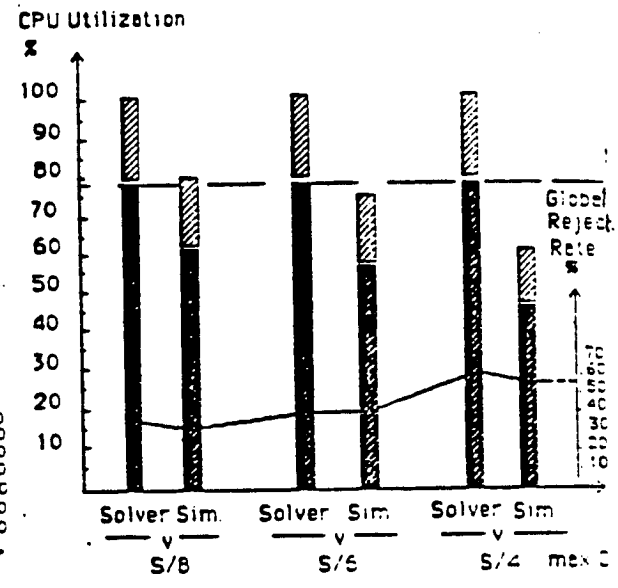


Figure 6.2: CPU Utilization with G_{78}

Figure 6 : Analytic Model velocity Domain CPU Utilization comparison.

max $C_{\emptyset}$		$P_{\emptyset}$	Layer service capacity $C_a(S)$	Request minimum execution time
S/8	Solver Sim	1.125•S .458•S	88 73	1.125•S 1.374•S
S/6	Solver Sim	1.166•S .499•S	86 67	1.166•S 1.479•S
S/4	Solver Sim	1.250•S .587•S	80 57	1.250•S 1.749•S

Table 3 : Periods, layer service capacity and request minimum execution time in the analytic model and in the simulator

Indeed, such a choice lets predict better performances with the solver, for which request minimum execution time is

$$X_{AM} = S + \max C_{\emptyset}$$

than in the simulator, for which, given three layers,

$$X = 3 \cdot (S/3 + \max C_{\emptyset}) = S + 3 \cdot \max C_{\emptyset}.$$

Experiments were carried on, in which the analytic model service period was set equal to the simulation request minimum execution namely :

$$P'_{AM} = S + 3 \cdot \max C_{\emptyset}$$

but the result was a disaster in terms of performance, rejection rates growing up to 50 % and more. This is easy to explain : The longest the period, the longest the waiting times in the arrival queue. And in this last experiment, the mean waiting time in the solver is three times that of the simulator's.

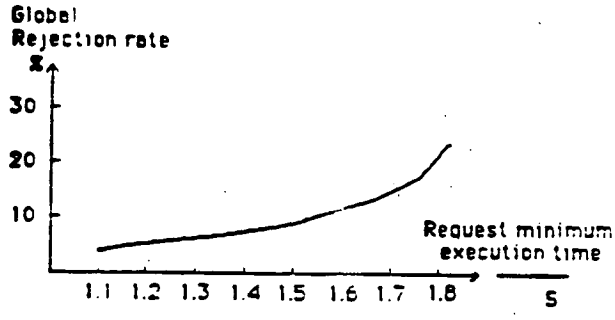


Figure 7 Rejection rates versus request minimum execution time for G_{78}

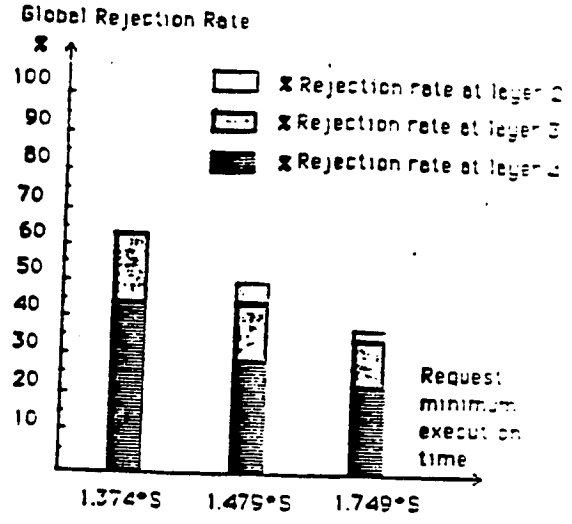


Figure 8 Rejection rates versus Request minimum execution time with G_{78} and simulator

Conclusions of this is that when load is in the order of 40 % and request maximum allowed execution delays are only about twice the request minimum execution time, the one file analytic model cannot be used to predict rejection rates.

. with G_{78}, the analytic model layer service capacity is always higher than the load whereas the simulator is always lower (Table 3). On the other hand, on Figure 6.2, we observe that, in the analytic model, CPU utilization for layer service is equal to the load whereas it remains lower in the simulator. This means that, in the simulator, promptness controls reject part of the requests before their last service termination time. The longer the request minimum execution time, the earlier the rejections do occur as shown on Figure 8. But such "early" rejections give to requests remaining in queues a higher probability to be served before their deadline.

Using the analytic model with a service period equal to

$$P_{AM} = S + 3^* \max C_{\emptyset}$$

was also experimented and the solver produced rejection rates for above those of the simulator, which leads to the conclusion that.when service load exceeds layer service capacity, promptness controls play a major role that cannot be approached by the one file analytic model.

VI.4. Scheduling efficiency

Scheduling efficiency is substantiated by comparing the rejection rates it produces with those that are produced by FIFO queues management.

VI.4.1. Scheduling efficiency at either light load or withe large delivery time constraints

The solver is used. Two poisson generators are used. One generates so to say " critical " requests with mean interarrival time λ_c and authorized processing delay D_c. The secund generates so to say background requests with mean interarrival time λ_b and authorized processing delay D_b. Of course,

$D_b > D_c$.

Both type of requests have the same static priority, namely zero. The global mean interarrival time is and :

$$\frac{1}{\lambda} = \frac{1}{\lambda_c} + \frac{1}{\lambda_b}$$

Rejection probabilites p_c and p_b are calculated for both traffics.

Results are show is on Figures 9.1, 9.2, 9.3 and 9.4.

" FiFo " ans " Sched " curves indicate max (p_c, p_b) when requests are processed respectively according to FiFo or to their global priority. p_c curves indicate rejection rates when only critical requests are generated. One notes that for loads smaller than 15 %,

- Rejection rate on critical traffic alone remains lower than 1%

- Scheduling both critical and background requests according to their global priority results in max (p_c,p_b) not exceeding 1.2 %, whereas FiFo management of both critical and background requests results in max (p_c, p_b) ranging within 2.5 and 5 %.

Such calculations are evidently prohibitiv in terms of run time with the simulator.

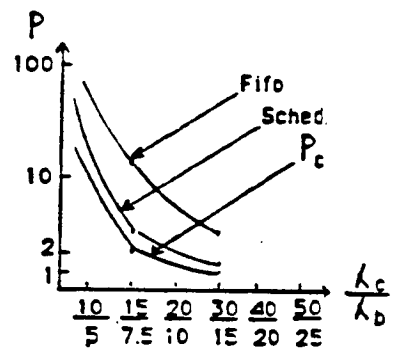


Figure 9.2 D_c=5ms, D_b=8ms
$\lambda_c = 2\lambda_b$

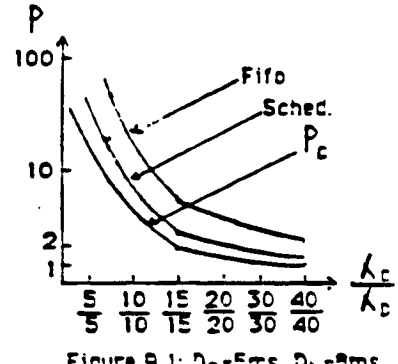


Figure 9.1 D_c=5ms, D_b=8ms
$\lambda_c = \lambda_b$

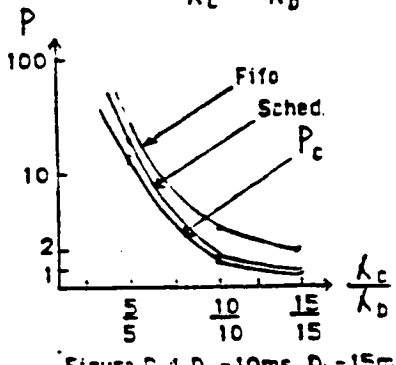


Figure 9.4 D_c=10ms, D_b=15ms
$\lambda_c = \lambda_b$

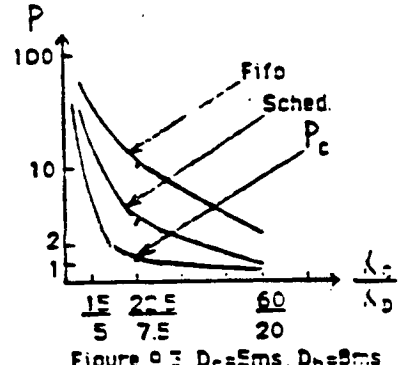


Figure 9.3 D_c=5ms, D_b=8ms
$\lambda_c = 3\lambda_b$

Figure 9: Rejection rates for light loads or large delivery times

VI.4.2. Scheduling efficiency at heavy loads and hard delivery time constraints.

Traffics G_{38} and G_{78} are used. The three layers model is used. Simulations are run with two options for both scheduling and FIFO queue management :

- Option 1 : Scheduling as well as FIFO management can occur periodically or be event driven. Periods values are those defined in experiments 2. In the event driven option, scheduling (or queue management) occurs after each service and upon request arrival if the arrival queue is empty.
- Option 2 : Promptness controls can be either validated or inhibited.

Scheduling options	R	R1	R2	R3	U(S)	U(Ø)
FIFO-p	20	36	0	0	38	2.7
Sched-p	16	26	4.5	2.6	38	8.2
FIFO-p + cp	16.2	29.9	0	0	36	2.7*
Sched-p + cp	12.5	24.7	0	.4	37	9.8
FIFO-evt	5.2	9.3	0	0	38	4
Sched-evt	3.9	7.5	.2	.2	38	4.9
FIFO-evt + cp	6.9	12.6	0	0	38	4*
Sched-evt + cp	2.7	5.4	.4	.4	38	6
Static priority		*0*	*1*	*2*		

Table 4.1 : Scheduling efficiency with G38

R : global rejection rate
R(g) : rejection rate for traffic g
U(S) : CPU utilization percentage for layer service
U(Ø) : CPU utilization percentage for scheduling or FIFO management
-p : periodic
+ cp : with promptness control
-evt : event driven

* In the case of FIFO, U(Ø) accounts only for context switching, excluding promptness control overheads for implementations reasons.

Scheduling options	R	R1	R2	R3	U(S)	U(Ø)
FIFO-p	100	100	100	100	72	6
Sched-p	100	100	100	100	72	19
FIFO-p + cp	37	67	0	0	72	6*
Sched-p + cp	32	38	23	25	72	17
FIFO-evt	30	52	2	3	78	6
Sched-evt	41	44	39	40	78	11
FIFO-evt + cp	22	40	0	0	70	5*
Sched-evt + cp	21	28	14	11	70	12
Static priority		*0*	*1*	*2*		

Table 4.2 : Scheduling efficiency with G_{78}

Table 4 : Scheduling efficiency compared to FIFO management's

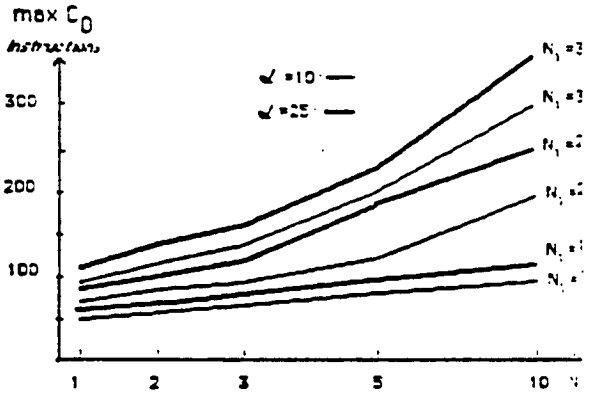
Results are reported on Tables 4.1 and 4.2. Promptness controls efficiency is obvious on either FIFO management and scheduling. But in the case of scheduling, rejections are distributed as most as possible on traffics with lower priority, that is g2 and g3, whereas in the case of FIFO management, rejections affect nearly exclusively traffic g1 which prevails in terms of arrival rate.

VII. CONCLUSION

An analytic model and an event driven simulator have been described that enable performance evaluation in the domain of real-time message scheduling within a transmission equipment designed according to the SO OSI-RM layered architecture. Both analytic model and simulator contain a scheduling policy based on message deadlines and static priorities. They differ however in the fact that promptness controls occur in the simulator and not in the model. It is proved that promptness control, that is, obsolete message rejection at each layer, is very efficient in minimizing messages rejection rates. Thus, whenever either traffic loads or message deadlines are very stringent in regards of layer service times, performance evaluation can only be accurate with simulation; when none of these two constraints is significant, the analytical model produces correct results with very short runs compared to simulations.Thus, both tools constitute an efficient platform kernel for real-time local area network evaluation. Of major interest is the fact that the simulator can be easily extended so as to emulate any protocol complexity and any traffics paterns.

Sched mode ↓ Loeu	P. C.	U(C) %	C_0 /instruction mean	C_0 /instruction σ	Queue size mean	Queue size σ	Queue size max
Period 38%	-	5 8	31	19	.3	45	6
	+	7	32	20	.3	.4	4
Event Driven 38%	-	2.8	22	13	.1	.5	3
	+	3 76	30	19	.1	20	4
Period 78%	-	13 64	62	20	84 (AC) .6 (SC)	49 .7	216 6
	+	12.2	51	22	1.3(AC) .5(SC)	1.5 .5	9
Event Driven 78%	-	6.1	30	17	3	4	23
	+	6	30	19	.3	45	4

Table 5 Meesured scheduler execution times for $N_1=3$, $W=1$, $\alpha=25$, $\beta=3$ and $\gamma=4$



$\max C_0 = W * (N_1 * (\alpha + \log_2 W * N_1) + \beta \log_2 \max Q - \beta)$

α = 10 ou 25, $\beta = 3$, $\gamma = 4$ instructions, $\max Q = 5$

Figure 10 Scheduling algorithm max execution time

REFERENCES

[IEE 84] IEEE 802.3 Standard for Carrier Sense Multiuple Access with Collision Detection, December 1984.

[IEE 85] IEEE 802.4 Standard for Token-Passing Bus Access Method, Feburary 1985

[DUR 85] Y. Durand, J.M. Pic, "Introduction to LAN design on Ariane 5 and Futrue Launchers", RTLAN Seminar, Bandol, France, INRIA Publish, April 85.

[ISO 81] "Data Processing Open Systems Interconnection Basic Reference Model", ISO/DP/7498

[KNU 73] D. KNUTH : " The art of computer programming : sorting and searching ", Vol.3, Addison Wesley Ed, 1973.

[LEL 85] G. Le Lann, J.F. Meyer, A. Movaghar, S. Sédillot, "Real-Time Local Area Network : Some Design and Modeling Issues", Rapport de Recherche INRIA 448, Oct. 85.

[MAP 85] R. KEIL, "MAP Specification", General Motors Technical Center, March 31, 1985.

[MIL 86] K. MILLS, M. Wheatley, S. Heatley, "Prediction of Transport Protocol Performance through Simulation", RTLAN Seminar, Bandol, France, April 86.