Accepted to be included in the proc of the 4th annual IEEE Conference on Automation Science and Engineering, Washington, USA, 2008.

Modeling and Response Time Evaluation of Ethernet-based control Architectures using Timed Event Graphs and Max-Plus Algebra

Boussad. Addad, Saïd. Amari

Abstract—**In this paper, a new approach to evaluate the response time in Ethernet-based automation systems using client server protocols, is presented. It is based on modeling the behaviour of the system using timed event graphs and the resulting state representation in Max-Plus algebra. First, an algorithm for tracking the frames in the architecture and giving the response time relative to any occurring event is explicated. Subsequently, analytical formulas for direct calculus of this delay are obtained. Finally, experimental measurements taken on a laboratory platform are used to check the validity of the method. Hence, the interest and effectiveness of our results become obvious. They can be used a posteriori to assess the delays in an existing architecture or a priori during the design phase to fulfill the time requirements of a control system.**

I. Introduction

Ethernet as a fieldbus in automation architectures is an attractive alternative. Indeed, it offers many advantages either economic with low costs and interoperability or technical with high throughputs reaching many gigabytes per second. Moreover, since the introduction of switches instead of hubs, the problem of collisions due to the medium access method CSMA/CD is solved. This protocol is non determinist and the frames can be affected with variable and even infinite theoretical delays. An unacceptable constraint that prevented the use of Ethernet at factory level even it has well established itself at the organization level of companies. Nowadays, Ethernet is more and more used in automation architectures and many suppliers developed their own protocols. Generally, all the solutions with advantages and disadvantages are more adequate for a specific application and it is difficult to put them in rank order [1]. In this study, we are interested in switched Ethernet networks that use the client server protocols like Modbus TCP/IP. It is a simple open standard protocol and overall compatible with standard Ethernet networks. It is an important feature that allows communication between the standard components of the architecture. Therefore, the integration of high level functions like monitoring or diagnostics in the control system is made easier. So far, such solution seems to be very satisfactory for many control applications. However, the presence of switches in the network makes it difficult to evaluate the architecture performances. Because of the arrival of parallel flows to the shared resources, it is not obvious to predict the delay a message suffers in a node of the network. Different works are made to assess these delays by the use of network calculus with worst case policy [2] and simulation [3]. Like the majority of studies, these works focus on the end-to-end delay of the network and ignore both the PLCs (programmable logic controllers) and the RIOMs (remote input output modules). The fact that the modules of the PLC are not synchronized and RIOMs shared by many applications, leads to considerable delays that have to be taken into account. To our knowledge, the studies that consider the whole architecture are so far, often based on simulation or direct measurements [4]. For instance, a method based on exhaustive state space exploration and model-checking [5]-[6]-[7], is used to assess the maximal bound of delay. This method does not provide distribution of response time and the computing limits are quickly reached because of the state explosion problem. Another approach relies upon modeling using hierarchical timed colored Petri nets and simulation with CPNTools [8]-[9]. It provides good evaluation of the delays but is onerous of time and the critical states are not surely scanned.

In our case study, we propose an analytic method to assess the bounds and distribution of response time. This paper is organized as follows. We begin by giving some recalls about timed event graphs (TEGs) and Max-Plus algebra in section II. Then in section III.*A*, an architecture with one PLC and one RIOM is presented before to move on to a more complex architecture in section III.*B*. The TEGs model and the resulting Max-Plus equations are developed in sections III.*A.1*. After, an algorithm and analytic formulae for response time calculus are obtained in sections III.*A.2* and III.*A.3*. Finally in section IV, measurements taken on the laboratory platform PRISME [10] are used to check the validity of the new approach we present in this work.

II. Timed event graphs and max-plus algebra

An event graph is an ordinary Petri net where all the places have at most one upstream and one downstream transition. An event graph is timed if the transitions or the places are affected with delays. We note n the number of transitions with at least one place upstream and m the number of source transitions t_u. The only place relying the transitions t_j and t_i is noted p_{ij} and its delay τ_{ij}.

To study the dynamic behaviour of TEGs, we assign each

Manuscript submitted March 20, accepted May 30, 2008.

B. Addad and S. Amari* are with Laboratory of Automated Production LURPA, ENS-Cachan/UniverSud/Univérsité-ParisXIII*, 61 av. du Président Wilson, 94235 Cachan Cedex, France (phone: +33-147402948; e-mail: {surname.name}@lurpa.ens-cachan.fr).

transition the date of its firing for k^{th} time. It is noted $u_i(k)$ for a source transition and $\theta_j(k)$ for others.

Example (Fig. 1):

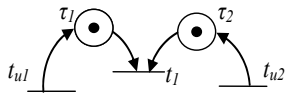

Fig. 1. Example of TEG.

The timed graph of the example leads to the equation:

$$\theta_1(k) = \max(\tau_1 + u_1(k-1), \tau_2 + u_2(k-1)), \quad (1)$$

which is a linear equation in the Max-Plus algebra. Indeed, a new algebraic structure emerged around two laws. The classical max noted indifferently in our study "max or $\oplus$" with a neutral element $\varepsilon = -\infty$ and the classical addition noted "+ or $\otimes$" with a neutral element $e = 0$. More details are available in [11]. So, the equation (1) can be rewritten:

$$\theta_1(k) = (\tau_1 \otimes u_1(k-1)) \oplus (\tau_2 \otimes u_2(k-1)). \quad (2)$$

In general, the behaviour of a TEG can be expressed using the following Max-Plus linear equations system:

$$\theta(k) = \bigoplus_{\varphi \geq 0} (A_\varphi \otimes \theta(k-\varphi) \oplus B_\varphi \otimes u(k-\varphi)), \quad (3)$$

where the vectors $\theta(k)$ and $u(k)$ components are the firing times of the transitions of the system for the k^{th} time. An element $A_{\varphi,ij}$ of the matrix A_φ represents the delay τ_{ij} associated to the place p_{ij} (with the marking φ) if it exists and ε else. Similarly for B_φ, it contains the delays of the places downstream the source transitions. In an analogous manner as in usual linear systems, this form can be brought to a state representation by replacing all the places with markings $\varphi_{ij} > 1$ by φ_{ij} other places and $(\varphi_{ij} - 1)$ intermediate transitions. Hence, we obtain an extended system with a state vector $x(k)$ described by the equation:

$$x(k) = \hat{A}_0 \otimes x(k) \oplus \hat{A}_1 \otimes x(k-1) \oplus \hat{B} \otimes u(k), \quad (4)$$

Can be rewritten in an explicit form:

$$x(k) = A \otimes x(k-1) \oplus B \otimes u(k), \quad (5)$$

where $A = \hat{A}_0^* \otimes \hat{A}_1$, $B = \hat{A}_0^* \otimes \hat{B}$ and $\hat{A}_0^* = \bigoplus_{i \in \mathbb{N}} \hat{A}_0^i$ is the Kleene star of $\hat{A}_0$. The last formulations permit to point out that the behaviour of a TEG is determinist, depending only on the source transitions and the initial conditions [12].

III. System Modeling and Response Time Evaluation

In our study, the automation architecture works according to client server protocol. The communication module of the PLC is the client and the RIOMs are the servers. The considered PLC is modular, with a CPU module (central processing unit) to execute the user program and an Ethernet board (ETHb) to send requests (combined requests: read and write data) to the RIOMs. Both modules work cyclically but without synchronization. The CPU accomplishes periodically the tasks: reading inputs, user program execution and updating outputs. In parallel, the ETHb sends requests to the RIOMs and waits for the answers. Once all the answers arrived, it stays waiting until the cycle time elapses to begin a new cycle. In our study, frames loss is not considered and no time-out has to be taken into account.

In this paper, two main cases are considered. A first architecture with one PLC and one RIOM as in Fig. 2 and a more complex one is studied later in the other case.

A. Case 1: one PLC and one module RIOM

The response time is the delay between the occurrence of an event on the plant and the arrival of the reaction event issued from the controller, on the process (Fig. 2). Two cases are to be considered in control systems. It may be, for instance, the delay between the detection of a danger and the triggering of an alarm. In this case the evaluation of the maximal bound of reaction time is of top priority. However, if the control concerns for example the speed of a motor, it is distribution rather than bounds of the response time that is more important to assess. In our work, we consider the general case, regardless of the events consequences.

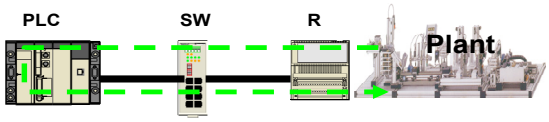

Fig. 2. Automation architecture (case 1).

1) TEGs Model and Max-Plus linear equations

According to the client server protocol described previously, we got to the architecture model of Fig. 3. It comprises two independent TEGs: one at the left that models the CPU and the other at the right for the rest of the system. This model enables only to know the current step of the cyclical applications. It makes abstraction of the meaning of the tokens in circulation in the TEGs. Indeed, the link between the CPU and the ETHb is hidden. It is clarified later by the fusion of the equations of the two TEGs. So, our modeling is firstly graphical and subsequently analytical.

The places p_1, p_2, p_3, and p_4 with delays τ_1, τ_2, τ_3 and τ_4 of the CPU, model respectively the phases of waiting, user program execution during T_{CLC} (reading and writing included), CPU busy and finally CPU idle. So, we easily note the periodical operating of the CPU with cycle period τ_3 equal to T_{CPU}. Similarly, τ_{14} models the scanning period T_{SCN} of the ETHb and a token in p_{14} means it is busy during at least this period. The sending of a request starts by firing t_4 and ends by firing t_5, τ_6 or T_{EM} being the required time to send a frame. A token in p_{13} means the request is sent and the ETHb is waiting for the answer. The places p_7 and p_{12} model the switch due delays imposed to the sent requests and the returned answers. The separation of these places is made possible since the links are full duplex

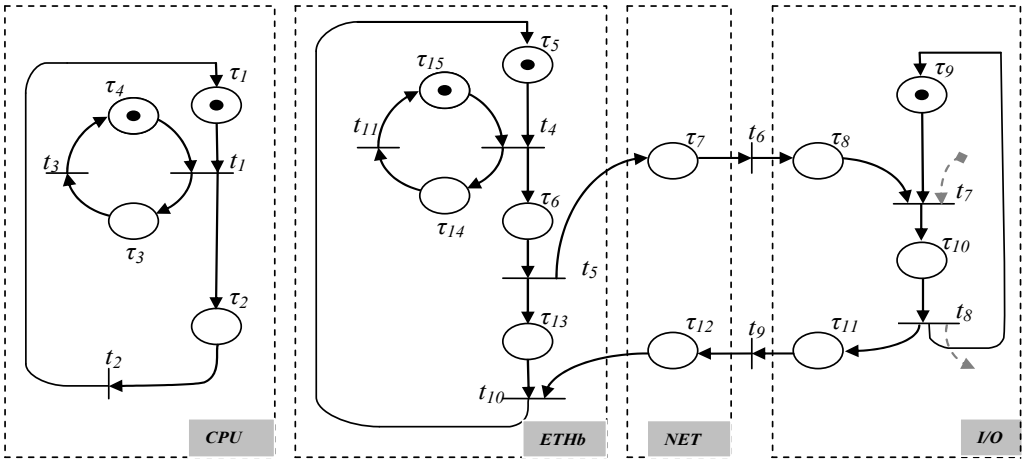

Fig. 3. TEGs model of the automation architecture.

and no situation of conflicts or collision is possible. To avoid overcrowding the scheme, τ_{12} includes also the necessary time to copy the response from the input buffer of the ETHb to the shared memory with the CPU (besides, this time is practically negligible). The places p_8, p_9, p_{10}, and p_{11} represent the RIOM. It stays waiting in p_9 until a request arrives to its input buffer p_8. By firing t_7, the processing starts and goes on for a time τ_{10} equal to $T_{I/O}$. At the end, it puts the answer in its output buffer p_{11}. The grey arrows represent the source (data coming from sensors) and output (data toward actuators). They are not considered since the system is not constrained and data are available at the output of the sensor as long as it is functional.

By applying the method of section II to the model with the initial conditions on Fig. 3, we got to the equations:

$$\begin{cases} \theta_1(k) = (\theta_2(k-1) \otimes \tau_1) \oplus (\theta_3(k-1) \otimes \tau_4) \\ \theta_2(k) = \theta_1(k) \otimes \tau_2 \\ \theta_3(k) = \theta_1(k) \otimes \tau_3 \end{cases} \quad (6)$$

$$\begin{cases} \theta_4(l) = (\theta_{10}(l-1) \otimes \tau_5) \oplus (\theta_{11}(l-1) \otimes \tau_{15}) \\ \theta_5(l) = \theta_4(l) \otimes \tau_6 \\ \theta_6(l) = \theta_5(l) \otimes \tau_7 \\ \theta_7(l) = (\theta_6(l) \otimes \tau_8) \oplus (\theta_8(l-1) \otimes \tau_9) \\ \theta_8(l) = \theta_7(l) \otimes \tau_{10} \\ \theta_9(l) = \theta_8(l) \otimes \tau_{11} \\ \theta_{10}(l) = (\theta_5(l) \otimes \tau_{13}) \oplus (\theta_9(l) \otimes \tau_{12}) \\ \theta_{11}(l) = \theta_4(l) \otimes \tau_{14} \end{cases} \quad (7)$$

The systems (6) and (7) are linear in Max-Plus algebra and can be rewritten in the form (4). We assigned them different indexes (k and l) to mean that they are not synchronized, exactly like the CPU and the Ethernet board. It is the additional and the main difficulty of this study.

2) Equations Resolution and simulation algorithm:

The resolution of the linear systems (6) and (7) leads to:

$$\begin{cases} \theta_1(k) = (k-1) \cdot T_{CPU} \\ \theta_2(k) = (k-1) \cdot T_{CPU} + T_{CLC} \\ \theta_3(k) = k \cdot T_{CPU} \end{cases} \quad (8)$$

$$\begin{cases} \theta_4(l) = (l-1) \cdot T_{SCN} \\ \theta_5(l) = \theta_4(l) + \tau_6 \\ \theta_6(l) = \theta_5(l) + \tau_7 \\ \theta_7(l) = \max(\theta_6(l) + \tau_8, \theta_8(l-1) + \tau_9) \\ \theta_8(l) = \theta_7(l) + \tau_{10} \\ \theta_9(l) = \theta_8(l) + \tau_{11} \\ \theta_{10}(l) = \max(\theta_5(l) + \tau_{13}, \theta_9(l) + \tau_{12}) \\ \theta_{11}(l) = \theta_4(l) + \tau_{14} \end{cases} \quad (9)$$

In these solutions, only the equations representing the following events interest us:

-- Reading and beginning of processing in the CPU (θ_1).

-- End of processing in the CPU and writing (θ_2).

-- Beginning of scanning and sending a request (θ_4).

-- Reception of an answer in the shared memory (θ_{10}).

Indeed, they are the events that link the CPU and the Ethernet board. When an answer arrives (θ_{10}), it is taken into account in the next beginning of the CPU cycle (θ_1). It is read and used in the calculus in the CPU. Once the processing finishes, the result is written in the memory of the Ethernet board (θ_2). It is taken into account on the next beginning of the scanning cycle (θ_4) and sent to the RIOM.

Let us put the time to wait for the reception of an answer: $T_r = \tau_6 + \tau_7 + \tau_8 + \tau_{10} + \tau_{11} + \tau_{12}$, then we obtain the following equations:

$$\begin{cases} \theta_1(k) = (k-1) \cdot T_{CPU} \\ \theta_2(k) = (k-1) \cdot T_{CPU} + T_{CLC} \end{cases} \quad (10)$$

$$\begin{cases} \theta_4(l) = (l-1) \cdot T_{SCN} \\ \theta_{10}(l) = (l-1) \cdot T_{SCN} + T_r \end{cases} \quad (11)$$

At the l^{th} scanning cycle, the answer is received at time $\theta_{10}(l)$ and to be taken in account, it has to wait for m^{th} (but the immediate next one with respect to $\theta_{10}(l)$) beginning of the processing cycle of the CPU. The condition to check is: $m = \underset{i/\theta_1(i)>\theta_{10}(l)}{Arg\min}(\theta_1(i)-\theta_{10}(l))$ where "Argmin" is the converse function which gives the index that minimizes the positive term: $(\theta_1(i)-\theta_{10}(l))$. Hence, we introduce a new variable $\hat{\theta}_1$ where: $\hat{\theta}_1(l)=\theta_2(m)=\theta_1(m)+T_{CLC}$.

As soon as the result of calculus is obtained, it is written in the memory of the Ethernet board. At the next scanning cycle, the n^{th} one, it is encapsulated in the request frame and sent to the RIOM. Again, we introduce another new variable $\hat{\theta}_2$ with the conditions: $\hat{\theta}_2(l)=\theta_4(n)$ and $n=\arg\underset{j/\theta_4(j)>\theta_1(l)}{\min}(\theta_4(j)-\hat{\theta}_1(l))$. The necessary time for the event consequence to get to the controlled process is therefore: $\hat{\theta}_f(l)=\theta_8(n)$. So, for the p^{th} event generated at time $v(p)$ and taken into account at the l^{th} scanning cycle, the associated response time of the architecture is given by:

$$D_r=\hat{\theta}_f(l)-v(p) \quad (12)$$

The delay is minimal if the data coming from the sensor are used in processing in the RIOM immediately after they are generated. So the minimal delay (not global minimum) for the p^{th} event is:

$$D_{MIN}=\hat{\theta}_f(l)-\theta_7(l)+d_f, \quad (13)$$

where d_f is the delay due to the data filtering in the RIOM. On the contrary, the delay is maximal if the data arrived immediately after the beginning of the RIOM processing relative to the previous scanning cycle. Thus, it is given by:

$$D_{MAX}=\hat{\theta}_f(l)-\theta_7(l-1)+d_f \quad (14)$$

This delay is always valid and it is the case if the frequency of update of the sensor output is smaller than the frequency of scanning. Else, some events may be erased and not used in any processing. This case is not considered in study even the formula (12) is not limitative and all what is necessary for calculus is the state of the system and the times of occurrence of the events.

Hence, we have an algorithm to evaluate the response time of the architecture relative to any occurring event. It is fast and easily implemented. The delays and the features of the components of the system are introduced as parameters. So, it is flexible for use for different configurations of the architecture. Simulations of this algorithm are used to check the validity of the formulas, obtained later in this study.

- *Lemma*

In the architecture, we have two cyclical applications but not synchronized. Despite of this constraint, the global system remains cyclical with a period T_{CR} that verifies: $T_{CR}=k_1\cdot T_{SCN}=k_2\cdot T_{CPU}$ where $k_1,k_2\in\mathbb{N}$ always exist.

Proof: Let us put: $T_{SCN}=r\cdot T_{CPU}$ with $r=[r]+\varepsilon$, $[r]$ being the entire part of r and ε fractional part. Since we can write $\varepsilon=n_1/n_2$, it is enough to take: $k_1=n_2$ and $k_2=n_2\cdot[r]+n_1$ to get to: $T_{CR}=(n_2\cdot[r]+n_1)\cdot T_{CPU}$ (14)
This period is minimal if n_1 and n_2 are prime numbers.

We can conclude that the algorithm is formal and all possible states of the system are scanned if the simulation length is over the period T_{CR}.

3) Analytical Calculus of response time

We use the results (9), (10) and the principle of the algorithm. Let us put: $T_r=\alpha\cdot T_{CPU}+\tau_r$, $T_{CLC}=\beta\cdot T_{CPU}$ where $\beta<1$. α and τ_r are respectively the ratio and the remainder of euclidean division of T_r by T_{CPU}.

For calculus complexity proven after, we begin with the case $r\in\mathbb{N}$ and generalize later (recall: $T_{SCN}=r\cdot T_{CPU}$).

- Case: $r\in\mathbb{N}$ ($\varepsilon=0$)

At the l^{th} scanning cycle, we have:
$$\theta_{10}(l)=(l-1)\cdot r\cdot T_{CPU}+\alpha\cdot T_{CPU}+\tau_r \quad (16)$$
For $k-1=(l-1)\cdot r+\alpha+1$ we obtain:
$$\theta_1(k)=\theta_{10}(l)+T_{CPU}-\tau_r \quad (17)$$
Since $0<T_{CPU}-\tau_r<T_{CPU}$ then:
$$\hat{\theta}_1(l)=\theta_1(k)+T_{CLC}=[1+\alpha+\beta+(l-1)\cdot r]\cdot T_{CPU} \quad (18)$$
We have also: $\theta_4(n)=(n-1)\cdot r\cdot T_{CPU}$ and for $n=l+1$ then:
$$\theta_4(n)=l\cdot r\cdot T_{CPU}, \quad (19)$$
Rewritten: $\theta_4(n)=\hat{\theta}_1(k)+[r-(1+\alpha+\beta)]\cdot T_{CPU}$ (20)
Thus on the condition C_1: $r>(1+\alpha+\beta)$, (it is the optimal case), $\hat{\theta}_2(l)=\theta_4(n)=\theta_4(l+1)$. This implies:

$$\begin{cases}\hat{\theta}_f(l)=\theta_8(l+1)\\ D_{MIN}=\theta_8(l+1)-\theta_7(l)+d_f\\ D_{MAX}=\theta_8(l+1)-\theta_7(l-1)+d_f\end{cases} \quad (21)$$

Finally:

$$\begin{cases}D_{MIN}=T_{SCN}+T_{I/O}+d_f\\ D_{MAX}=2T_{SCN}+T_{I/O}+d_f\\ D_r(p)=\theta_8(l+1)-v(p)\end{cases} \quad (22)$$

In practice the condition C_1 is often respected and the results (22) are valid since the scanning period is by far longer than the period of the CPU. The results are very interesting since the calculated bounds are constant and therefore global extrema. Indeed, the network and the RIOM processing delays are constant. It is not the case in architectures with acyclic traffic. This is out of our study.

The results (22) are easily generalized for a condition: $r\cdot q>(1+\alpha+\beta)>r\cdot(q-1)$ and we found:

$$\begin{cases} D_{MIN} = q \cdot T_{SCN} + T_{I/O} + d_f \\ D_{MAX} = (q+1) \cdot T_{SCN} + T_{I/O} + d_f \\ D_r(p) = \theta_8(l+q) - v(p) \end{cases}, \quad (23)$$

- Case : $r \in \mathbb{Q}^+$ $(\varepsilon \neq 0)$

Let us put: $\tau_r = \gamma \cdot T_{CPU}$ (where obviously $\gamma < 1$).

At the l^{th} scanning cycle, we have:
$\theta_{10}(l) = (l-1) \cdot r \cdot T_{CPU} + (\alpha + \gamma) \cdot T_{CPU}$ (24)

Let us take $i \in \mathbb{N}$ where: $i \leq \gamma + \varepsilon \cdot (l-1) < (i+1)$ (*)

For $k - 1 = (l-1) \cdot \lceil r \rceil + \alpha + 1 + i$ we get:
$\theta_1(k) = \theta_{10}(l) + \left[i + 1 - (\gamma + \varepsilon \cdot (l-1)) \right] \cdot T_{CPU}$ (25)

and since in (*) $i \leq \gamma + \varepsilon \cdot (l-1) < (i+1)$, then:
$\hat{\theta}_1(l) = \theta_{10}(l) + \left[i + 1 + \beta - (\gamma + \varepsilon \cdot (l-1)) \right] \cdot T_{CPU}$. (26)

We have also:
$\theta_4(n) = (n-1) \cdot r \cdot T_{CPU}$ and for $n = l+1$ then:
$\theta_4(n) = l \cdot r \cdot T_{CPU}$ i.e.
$\theta_4(n) = \hat{\theta}_1(k) + \left[r - \left[\alpha + \beta + i + 1 - \varepsilon \cdot (l-1) \right] \right] \cdot T_{CPU}$. (27)

From (*), we deduce: $\gamma < i + 1 - \varepsilon \cdot (l-1) \leq 1 + \gamma$

Let us put: $\Gamma_{l,i} = i + 1 - \varepsilon \cdot (l-1)$ with $\gamma < \Gamma_{l,i} \leq 1 + \gamma$

$$\Gamma_{MIN} = \min_{i \in \mathbb{N}, l \in \mathbb{N}} (\Gamma_{l,i})$$

$$\Gamma_{MAX} = \max_{i \in \mathbb{N}, l \in \mathbb{N}} (\Gamma_{l,i})$$

Thus, on the condition C_2: $r > (\alpha + \beta + \Gamma_{MAX})$, we obtain :
$\hat{\theta}_2(l) = \theta_4(n) = \theta_4(l+1)$ i.e. $\hat{\theta}_f(l) = \theta_8(l+1)$. As we note, this leads to the same results as in (21). But if this condition is not respected, the bounds of time would depend on $\Gamma_{l,i}$. So, the conditions of bounds calculus are global (absolute) and the delay condition relative to the p^{th} event is local.

On the following global and local conditions:
$r \cdot q_1 > (\Gamma_{MIN} + \alpha + \beta) > r \cdot (q_1 - 1)$
$r \cdot q_2 > (\Gamma_{MAX} + \alpha + \beta) > r \cdot (q_2 - 1)$,
$r \cdot q_3 > (\Gamma_{l,i} + \alpha + \beta) > r \cdot (q_3 - 1)$
the global bounds of time and local delay are given by:

$$\begin{cases} D_{MIN} = q_1 \cdot T_{SCN} + T_{I/O} + d_f \\ D_{MAX} = (q_2 + 1) \cdot T_{SCN} + T_{I/O} + d_f \\ D_r(p) = \theta_8(l + q_3) - v(p) \end{cases}. \quad (28)$$

In this general case, we point out that the optimality condition $r > (\alpha + \beta + \Gamma_{MAX})$ is more restrictive than in case $r \in \mathbb{N}$. Indeed, for $\varepsilon = n_1 / n_2$, it is enough to take $(l-1) = n_2$ and $i = n_1$ to obtain $\Gamma_{l,i} = 1$. This implies $\Gamma_{MAX} \geq 1$ and the condition C_2 more restrictive than C_1.
It is an important result which suggests to set the period of scanning as a multiple of the period of the CPU (of course minimize T_{CPU} first), in order to reduce the maximal bound of response time of the architecture.

B. Case 2: one PLC and N RIOMs

The modeling of the PLC does not change and only the number of RIOMs that is different. The requests are sent from the Ethernet board in an invariant order and the switch with FIFO policy is without quality of service. In this more general case, the TEG of the CPU remains the same but the other one becomes more complex. We introduce other equations to model the FIFO policy and solve the problem of sharing resources: the switch and the Ethernet board. The RIOMs are affected with indexes according to the order of their scanning. We associate the index i to the RIOM receiving the i^{th} request from the Ethernet board. Particularly, N_S and N_D are the indexes assigned to respectively the event source (S) and destination (D) of its consequence. On Fig. 4 for example, $N_S = 4$ and $N_D = 5$.

With similar analysis and same notations as in the first case with one RIOM, the following results are obtained:

$$\begin{cases} D_{MIN} = q_1 \cdot T_{SCN} + (N_D - N_S) \cdot T_{EM} + \tau_7^D - \tau_7^S + T_{I/O}^D + d_f \\ D_{MAX} = (q_2 + 1) \cdot T_{SCN} + (N_D - N_S) \cdot T_{EM} \\ \qquad\qquad + \tau_7^D - \tau_7^S + T_{I/O}^D + d_f \end{cases} \quad (29)$$

where: $r \cdot q_1 > \left[\min_{l,i} (\Gamma_{l,i} + \alpha(l)) + \beta \right] > r \cdot (q_1 - 1)$

$$r \cdot q_2 > \left[\max_{l,i} (\Gamma_{l,i} + \alpha(l)) + \beta \right] > r \cdot (q_2 - 1).$$

$T_r(l) = max(N \cdot T_{EM}, N_S \cdot T_{EM} + \tau_7^S + \tau_{12}^S + T_{I/O}^S)$, (30)
$T_r(l) = (\alpha(l) + \gamma(l)) \cdot T_{CPU}$, is the necessary time to wait for the reception of the answer from the event source.
$T_{I/O}^S$ and $T_{I/O}^D$ are processing times of RIOMs (S) and (D).
τ_7^S, τ_{12}^S and τ_7^D are the network delays imposed respectively to the request going to the event source, the response coming from the event source and the request going to the event consequence destination. For a given architecture with RIOMs with constant processing times, these delays are invariant and we have to calculate them only once. The scanning is cyclical and they always arrive to the switch or the Ethernet board in the same order.

The results are very interesting and to get small response time, we should assign great index to the source and small one to the destination: the order of RIOMs is important. However, we have to keep in mind that the condition of calculus of the delay depends on $T_r(l)$ or $\alpha(l)$ and we should decrease N_S (see (30)). So, the optimal case is got by increasing N_S and stop just before the condition changes.

IV. VALIDATION OF THE MODEL AND THE METHOD

To check the validity of the model and the results developed previously, we consider two configurations of architectures (Fig. 2 and Fig. 4). We compare the results of the algorithm and the formulas with experimental measurements taken on the patented platform PRISME [10].

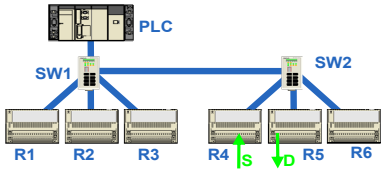

Fig. 4. Automation architecture (case 2).

In the second configuration, we are interested in the causality delay between an event generated on the input of the RIOM R4 and its consequence on the output of the RIOM R5. The histograms of Fig. 5 represent a series of 10,000 measurements and simulations of the algorithm for this configuration. This architecture is more general than those of the study (two switches). This is made to show the possibility to extend the results to more complex systems.

The CPU period is set up to 5 ms and scanning to 10 ms with a jitter of 15%. The jitter is considered in the algorithm and the formulas. We obtained the results of the table (I).

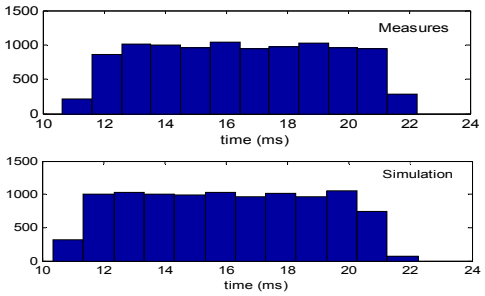

Fig. 5. Histograms of measured and simulated delays (case 2)

TABLE I
RESULTS OF MEASUREMENTS, SIMULATIONS AND FORMULAS

		Response delays in ms		
		Min	Max	Mean
Case 1	Measures	10.40	21.90	16.10
	Simulation	10.06	22.24	15.82
	Formulas	10.06	22.24	/
Case 2	Measures	10.65	22.25	16.40
	Simulation	10.31	22.49	16.07
	Formulas	10.31	22.49	/

In both cases, we already can conclude about the validity of the formulas because the max delays are greater than all others and the min delays are smaller than all. As expected, the results of the simulation and formulas are exactly the same in all cases. Indeed, they are based on the same principle. The gaps of delays, with respect to measurements, are in all cases smaller than 3.27% for analytical formulas and simulations. This gap is in both cases, smaller than 2.01% in the calculus of the mean of responses times. A random event generator is used in simulation to obtain realistic distribution of delays (to offset effects of the jitter). Thus, the shapes of the measurements and simulations histograms are very similar (Fig. 5).

If we compare the results of both configurations, we note that there is about a difference of 0.25 ms between the maximal bounds (also for minimal bounds). It is exactly the value of transmission time of a frame T_{EM} and since the switches are very fast, the considered configurations are very similar. The main difference is the use of a RIOM for event source and another for consequence destination with a difference of one in order (R4 and R5). This result consolidates the general formulas of section III.*B*.

V. CONCLUSION

In this work, we presented a new approach to evaluate time performances in Ethernet automation architectures using client server protocols. A formal algorithm and analytical formulas for trivial evaluation of the response time are developed. The comparison of the results with experimental measurements, allowed us to check the validity of both the algorithm and the formulas. Thus, by the use of these interesting formulae, it is easy to choose the adequate configuration of the components of an architecture to fulfil the desired requirements. For further studies, it would be interesting to consider more general automation architectures with many PLCs and acyclic traffic. Finally, a study of a control system over an Ethernet network with quality of service is prospected.

REFERENCES

[1] P. Neumann, "Communication in industrial automation -what is going on?" Control Engineering Practice, 2006.

[2] J. P. Georges, E. Rondeau, and T. Divoux, "Evaluation of switched Ethernet in an industrial context using network calculus," in *Proc. of 4th IEEE Int. Workshop on Factory Communication Systems*, 2002.

[3] N. Kakanakov, M. Shopov, G. Spasov and H. Hristev, "Performance evaluation of switched Ethernet as communication media in controller networks," *International Conference on Computer Systems and Technologies (CompSysTech'07)*, pp. IIIA.8-1-6, 2007.

[4] K. C. Lee and S. Lee, "Performance evaluation of switched Ethernet for real-time industrial communications," *Computer standards & interfaces*, (24):411–423, 2002.

[5] J. Greifeneder, G. Frey, "Optimizing quality of control in networked automation systems using probabilistic models," in *Proc. of 11th IEEE Int. Conf. on Emerging Technologies and Factory Automation*, Prague, Czech Republic, September 2006.

[6] B. Ben-Hédia, F. Jumel and J.-P. Babau, "Formal evaluation of quality of service for data acquisition systems," in *Proc. of FDL'05*, 2005.

[7] D. Witsch, B. Vogel-Heuser, J. Faure, and G. Poulard-Marsal, "Performance analysis of industrial Ethernet networks by means of timed model-checking," in *Proc. of 12th IFAC Symposium on Information Control Problems in Manufacturing*, pp 101–106, 2006.

[8] D. A. Zaitsev, "Switched LAN simulation by colored Petri nets," *Mathematics and Computers in Simulation*," 65(3):245–249, 2004.

[9] G. Marsal, B. Denis, J.-M. Faure, G. Frey, " Evaluation of response time in Ethernet-based automation systems," in Proc. of 11th IEEE Int. Conf. on Emerging Technologies and Factory Automation, Prague, Czech Republic, September 2006.

[10] B. Denis, O. De Smet, J.-J. Lesage, J.-M. Roussel," Process of performance analysis and identification of systems as finite state automata", FR. Patent 01 110 933, August 2001.

[11] F. Baccelli, G. Cohen, G.-J. Olsder, and J.-P. Quadrat, *Synchronization and Linearity: An algebra for Discrete Event Systems*, Wiley, 1992.

[12] F. Baccelli, G. Cohen, and B. Gaujal, "Recursive equations and basic properties of timed Petri nets," *Discrete Event Dynamic Systems: Theory and Applications*, 1 (4), 1992.