Measuring the impact of vertical integration on response times in Ethernet fieldbuses

Bruno Denis, Silvain Ruel, Jean-Marc Faure, Gaëlle Marsal, Georg Frey

Measuring the impact of vertical integration on response times in Ethernet fieldbuses

Bruno Denis, Silvain Ruel, Jean-Marc Faure
LURPA, ENS Cachan, UniverSud Paris
61 av. du Pdt Wilson
94235 Cachan Cedex, France
{denis,ruel,faure}@lurpa.ens-cachan.fr

Gaëlle Marsal
EDF, R&D
6, Quai Watier – BP 49
78401 Chatou Cedex, France
gaelle.marsal@edf.fr

Georg Frey
University of Kaiserslautern
Erwin-Schrödinger-Str. 12
67663 Kaiserslautern, Germany
frey@eit.uni-kl.de

Abstract

The introduction of Ethernet and Internet technologies in the fieldbuses of automation systems widely facilitates vertical integration. Control functions can coexist more easily with higher level functions such as supervision, production reporting or maintenance. But what does the response time of the control function become when architecture components – e.g. controllers, remote input-output modules or fieldbuses – are requested in parallel by other functions?

In this paper, we focus on switched Ethernet-based fieldbuses using Modbus TCP/IP. The evaluation of the impact of vertical integration on the response time of control functions is obtained from a series of measurements on a real system. A great number of response times were measured with a specific automated equipment that we developed. Three types of load on the control architecture are studied: traffic on fieldbuses, requests on controllers and requests on remote input-output modules. The analysis of the measurement results shows which level of vertical integration can be reasonably allowed, taking into account the required response times.

1. Introduction

For a long time, the Ethernet technology has been used in factory networks [4], [10], but its introduction at the field level – to replace traditional fieldbuses such as Profibus DP or WorldFip – is recent [12]. Indeed, the characteristics of Ethernet are not originally dedicated to this role, and it is thus important to evaluate their impact on the time performances of an automation system.

The works devoted to this topic can be distinguished according to the kind of method used: either *a priori* or *a posteriori* analysis. The first kind of method aims to evaluate the performances of a system during the design phase, i.e. before it exists, and consequently it requires models of the system. The following approaches can be quoted: methods based on analytic models (Worst Case evaluation [5], Network Calculus [2]), exhaustive state space model exploration (timed model-checking [13], probabilistic model-checking [3]) and partial state space model exploration (simulation [9], [14]). The second kind of evaluation method aims to evaluate the performances of a system during its running phase, i.e. when it already exists, and consequently it consists of measurements [11], [6].

Most of these works deal with network features such as network cycle time, throughput, or transmission delay. Only a few of them deal with application level performances such as response time, i.e. the delay between the occurrence of an input event and the corresponding occurence of an output event [1]. Moreover, if control applications share the Ethernet fieldbus with higher level applications (vertical integration), the evaluation of control architecture performances becomes complex.

The work which is presented in this paper stands in the second category: *a posteriori* analysis. It aims to assess the impact of vertical integration on response time.

Section 2 presents in detail the automation architecture studied. In section 3, this architecture is extended so as to integrate high level automation functions. The experiments that we performed on this extended architecture are described in section 4 whereas the results obtained are presented in section 5. From these results, an analysis leading to advice for the organization of communication within automation architectures is performed in section 6.

2. Horizontal integration within Modbus TCP/IP fieldbuses

2.1. Architectures studied

Our work focuses on a particular class of architecture dedicated to the control functions (Fig. 1). These architectures are built around one Ethernet-based sub-network in which logic controllers and remote input-output modules (RIOM) work together to carry out control functions. The interaction of several control functions on one sub-net is called horizontal integration.

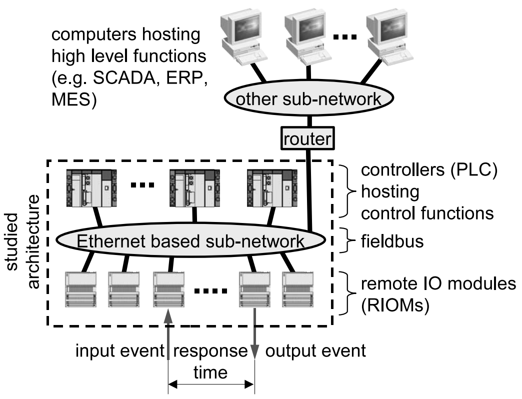

Figure 1. Structure of an Ethernet-based automation system

To characterise such architectures, we use response times as performance criteria. The following assumptions will be made for the components of the architecture class studied:

- Controllers – Programmable Logic Controllers or industrial computers – are modular. A PLC processor module is dedicated to control functions, whereas a communication module deals with communications. Thus, with this kind of controllers, two processors will work simultaneously and in an asynchronous way, one for control functions and the other one for communication functions.

- Inputs and outputs are gathered in remote input-output modules (RIOM) directly connected to the network.

- The network is Ethernet-based, mixing 10baseT and 100baseT technologies. It is composed of Ethernet switches and Ethernet cables.

- All control functions are hosted by PLC processor modules, all of them being located in the same Ethernet sub-network.

- All possible high level functions – e.g. SCADA, ERP, MES – are hosted by computers in other sub-networks connected to the fieldbus through routers.

- The selected protocol is open enough to facilitate inter-network communications. Modbus TCP/IP has been chosen in our case.

2.2. Modbus TCP/IP

Based on the *de facto* industrial standard Modbus for serial communication, Modbus TCP/IP is an application-layer messaging protocol[1], positioned at the level 7 of the OSI model, and relying on the TCP/IP stack. It provides client/server communication between connected devices. In our class of architecture, RIOMs are Modbus servers while PLC communication modules are clients. To perform control functions, Modbus TCP/IP is supplemented by an input-output cyclic scanning algorithm (IOscanning). According to [8], Modbus TCP/IP could be only considered as real time class 1, i.e. soft real time, because IOscanning acts on top of the TCP/IP stack. To be considered as real time class 2, i.e. hard real time, it is necessary to use scheduling of data traffic on top of the MAC layer.

As an open protocol, in particular with the use of the TCP/IP stack, Modbus TCP/IP is a good choice for an easy vertical integration.

2.3. Case study

The basis of our case study is a hardware control architecture including 2 modular controllers, 9 RIOMs and 3 Ethernet 8-port switches. All components are connected as shown in Fig. 2. The whole architecture is made of industrial components off-the-shelf from the Schneider-electric company (arbitrary choice): TSX 57203 for PLC processor modules, ETY 5102 for PLC Ethernet 100baseT communication modules, 170 ENT 11000 for Ethernet 10baseT adapters of RIOMs, 499 NES 18100 for Ethernet 8-port 100baseTX switches.

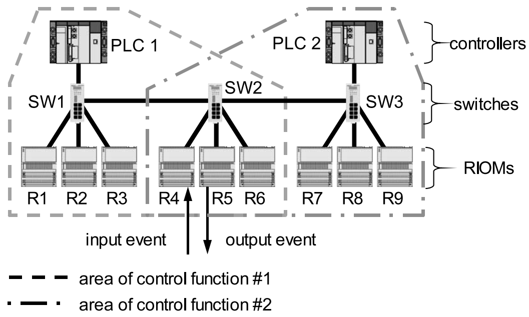

Figure 2. Basis of our case study

The software of the control architecture includes two control functions. Each one is implemented in one PLC processor module, one by controller, and computed cyclically in at most 3 ms. The input and output data of these functions are collected and updated periodically, every 10 ms, using the communication module of each controller. Function #1 (resp. #2) implemented into PLC1 (resp. PLC2) scans inputs and outputs from remote modules R1, R2, R3, R4, R5 and R6 (resp. R4, R5, R6, R7,

[1] http://www.modbus.org/

R8 and R9). The two functions share 3 modules (R4, R5 and R6) and one switch (SW2).Thus, these two functions interact in a horizontal integration.

Our former works in [7] have shown by using simulation that the horizontal integration of control functions in such architectures can preserve the real time characteristics in some cases. To illustrate the real time class of a network, the distribution of its cycle time is often presented. In our case study, two time cycles cohabit in the same subnetwork. Fig. 3.a presents the distribution of 10,000 measurements of the network cycle time of control function #1. Without vertical integration, our case study provides an average network cycle time of 10.00 ms with 15.01 % jitter. It has the characteristics of a real time class 2 network even if, in general, Modbus TCP/IP networks do not have this feature.

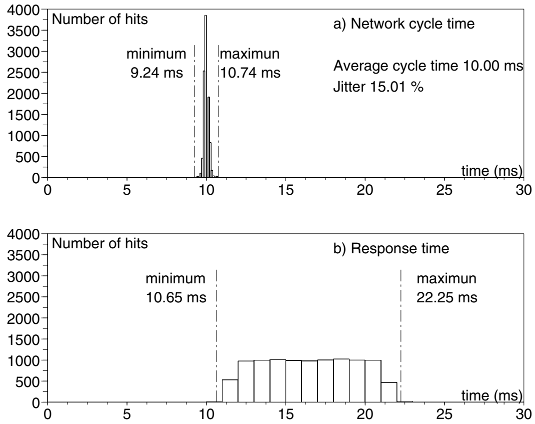

Figure 3. Time performances of case study basis – a) Network cycle time, top graph, b) Response time, bottom graph

The time performance retained in this case study is the response time of function #1, which will be measured as the delay between the occurrence of an input event on shared module R4 and the occurence of the corresponding output event on shared module R5 after treatment in PLC1. Fig. 3.b presents the distribution of 10,000 measurements of this response time. This is the reference distribution for the basis of the case study, i.e. without vertical integration.

3. Vertical integration and resource sharing

In automation systems, vertical integration appears when functions with higher level than control functions are added. SCADA, MES, ERP or maintenance functions are typical examples. In the context of Modbus TCP/IP networks, the introduction of these functions results in new requests to remote IO modules and/or controllers (Fig. 4). Thus, new traffics appear, resource sharing increases and the time performance of control functions could be affected.

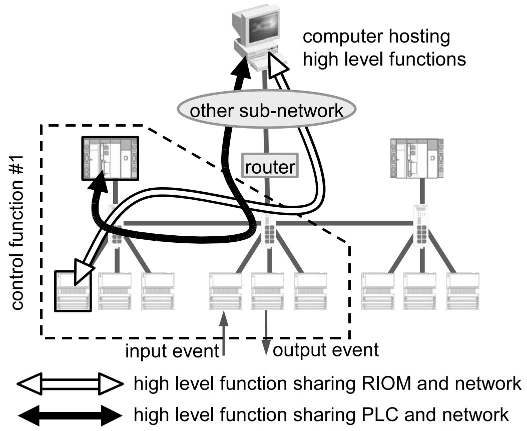

Figure 4. Vertical integration

To perform a more accurate analysis of the impact of vertical integration, we will excite each resource of our case study independently rather than excite it by using one of the two types of requests presented in Fig. 4. While following the path of these new requests, the following shared resources are identified: network resources (switches and cables), RIOMs (Modbus servers) and PLCs (Modbus clients for control functions, Modbus servers for higher level functions, and processors which perform functions of control).

To recreate a wide variety of vertical integration, the following parameters will be considered:

- To excite switches and cables, a flow of Ethernet frames is used, and the experiments parameters are the bit throughput and the frame size.
- To excite RIOMs and PLCs, a flow of Modbus requests is used, and the parameters are the number of concurrent clients to one RIOM or one PLC, and the periodicity of the requests.

4. Design of experiments

Knowing the resources involved in vertical integration, experiments must be designed to generate gradual resource loads and measure the impact on the response time. The following subsections aim to present how each kind of resource is loaded as independently as possible of the other ones, and how to generate the input event in order to measure the resource performances.

4.1. Network load generator

In our case study, 8-port Ethernet switches are used. Taking into account hardware architecture, three ports remains to add a load for each one of these switches. These free ports are used to connect computers with the traffic generator software Iperf[2]. Iperf allows to create UDP traffic between the computers with controlled throughput and controlled Ethernet frame size.

[2] http://dast.nlanr.net/Projects/Iperf/

To study the impact of the switch load on response time, the hardware architecture of the case study is supplemented with up to three computers connected to switch SW1 (Fig. 5). Each computer-switch full duplex link allows a maximal theoretical throughput of 2 x 100 Mbit/s. Thus, with three computers connected, it is possible to generate a traffic of up to 600 Mbit/s into the switch fabric.

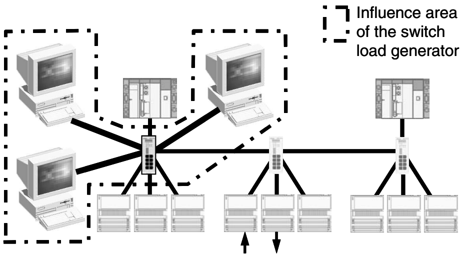

Figure 5. Load generator for one Ethernet switch

To study the impact of the Ethernet link load on time performance, the hardware architecture is supplemented with 2 computers with IPerf, connected respectively to switchs SW1 and SW2 (Fig. 6). This configuration allows to generate a maximal theoretical throughput of up to 200 Mbit/s into link SW1-SW2. It should be noted that this kind of load also implies a switch fabric load.

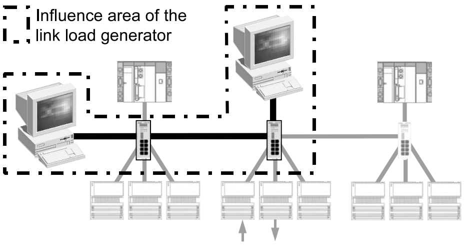

Figure 6. Load generator for one Ethernet Link

4.2. Remote IO module load generator

To study the impact of the RIOM load on time performance, the hardware architecture of the case study is supplemented with up to 9 computers – according to the 9 free switch ports in the architecture – uniformly distributed on the 3 switches (Fig. 7 presents a configuration with only one computer). Each computer periodically generates 9 Modbus TCP/IP requests, one for each RIOM. This is obtained thanks to a Python[3] script developed according to Modbus open specifications using the TCP/IP socket package. This script allows to setup the request period, whereas the type and size of the requests are always the same because of the RIOM capacity. A request consists of one 16-bit register reading and one 16-bit register writing using the read/write multiple registers Modbus function (code 0x17). This function performs a combination of one reading operation and one writing operation in a single Modbus transaction. It should be noted that this kind of load also implies a network load.

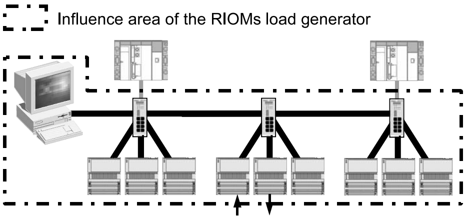

Figure 7. Load generator for RIOMs

The systematic excitation of the 9 RIOMs was chossen to avoid having to determine which loaded RIOM will have the most important impact.

4.3. Controller load generator

The load generation principle for a controller is the same as for load generation for RIOMs. However, only PLC1, which hosts function #1, is loaded (Fig. 8). The Python script allows to set up the request period, whereas the type and size of the request are always the same (read and write one 16-bit register). Using the same request type for RIOMs and controller load will allow to compare their behaviour under a vertical integration load.

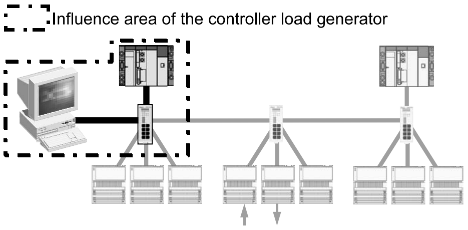

Figure 8. Load generator for one controller

4.4. Response time measurement

To measure the response time of function #1, a specific tool developed at LURPA was used. This tool named PRISME[4] is dedicated to experiments on Discrete Event Dynamic Systems. A square signal generator ensures the excitation of input RIOM, while a logical analyzer observes at the same time the input signal and the output RIOM reaction (Fig. 9). Thanks to a GPIB network, a computer automates the repetition of the following tasks: set-up of the square signal period in the generator, trigger acquisitions in the logical analyzer, download measurements, compute and save response times.

To obtain a good evaluation of the response time, it is necessary to make lots of measures with different delays between the occurence of the input event and the beginning of the IOscanning cycle in the PLC communication

[3] http://www.python.org

[4] French patent # 01 110 933

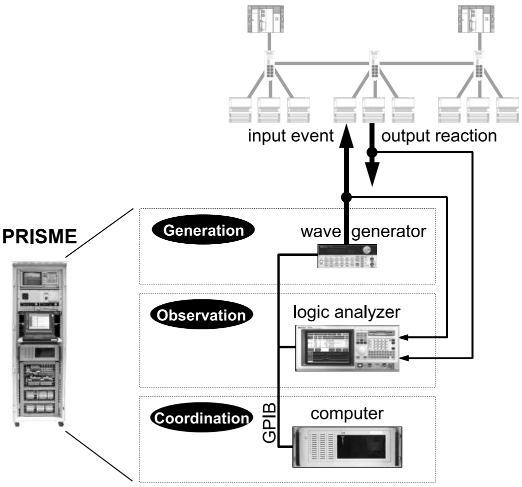

Figure 9. Response time measurement

processor (represented by delay δ on Fig. 10). At best, the distribution of these delays should be uniform.

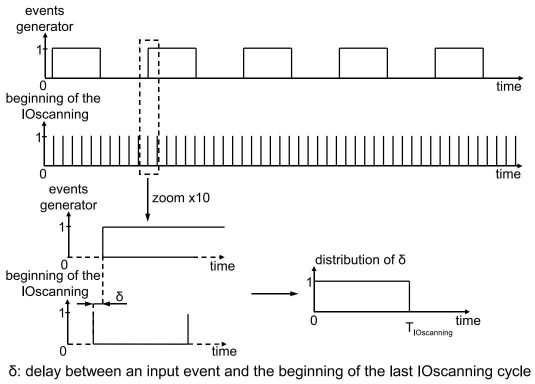
events generator
1
0
time
beginning of the IOscanning
1
0
time
events generator
zoom x10
1
0
time
beginning of the IOscanning
δ
1
0
time
distribution of δ
1
0
T_IOscanning
time
δ: delay between an input event and the beginning of the last IOscanning cycle

Figure 10. Timing diagram of smart event generator compared to IOscanning

Thus we define the input event period T_{input} (equation 1) as a function of the IOscanning cycle time $T_{IOscanning}$.

$$T_{input} = k * T_{IOscanning} + \frac{T_{IOscanning}}{100} \tag{1}$$

Parameter k is an integer which should be high enough to allow a complete propagation of the input event through the automation architecture before generating the next one. The second term, which is proportional with a hundredth of the IOscanning time cycle, allows a systematic sweep of this cycle. In our case, the IOscanning cycle time is set up to 10 ms, and we chose k equal to 10, thus the input signal period is 100.1 ms.

Thanks to PRISME tool, 350 measures of response time per minute are obtained automatically with a period of 100.1 ms for the input signal, and including the time to download and back up the data. The data are postprocessed with the Scilab open source platform[5] for numerical computation, to provide the graphs that are displayed in the next section.

5. Results obtained

Each experiment consists of obtaining a sample of 10,000 measurements of the response time of function #1 for a fixed set of load parameters. About 60 experiments were performed for the case study, corresponding to 30 cumulated hours of automatic acquisitions, and giving more than half a million of measurements. For each experiment, one response time histogram is postprocessed. Fig. 11 presents three of them. Fig. 11.a is the reference histogram of the case study (already presented Fig. 3.b with different scales) without load coming from vertical integration. In a first analysis, histograms which correspond to heavy loads on the network (Fig. 11.b) and to heavy loads on the controller (Fig. 11.c) show that vertical integration can have a significant impact on the response time.

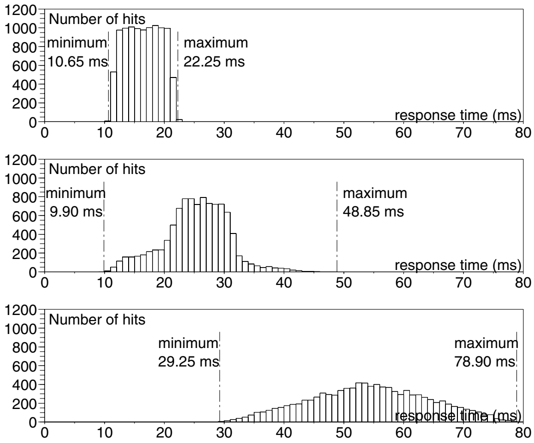

Figure 11. Response time – a) loadless, top graph, b) with heavy load on controller, middle graph, c) with heavy load on Ethernet link, bottom graph

The shape of histograms is interesting for qualitative analysis, but quantitative indicators are more relevant to facilitate the comparison between the experiments. Thus, in the histograms, the maximum and minimum response times measured are indicated. These two data are very useful for control architecture designers, because the first one provides the worst case, the second one provides the best case and their difference gives the range. The average value was not retained because it sometimes tends to minimize load effects by masking very high maxima. In the following, the maximum response times measured will be retained as a comparison criterion between measurements.

Let us examine the first two histograms in Fig. 11.a and Fig. 11.b more closely. The loaded configuration (Fig. 11.b) has a smaller minimum response time than the loadless configuration (Fig. 11.a). This result seems to be

[5] http://www.scilab.org/

paradoxical: an increase in load seems to involve a reduction in the minimum response time!

Actually, when a load is present in the control architecture, an input event can be taken into account with a delay in the IOscanning cycle. The two chronograms on top of Fig. 12 represent the activity of the PLC processor and PLC communication modules. The three timing diagrams in the middle of the figure describe a scenario giving a minimum response time in the loadless architecture. The three chronograms at the bottom of the figure describe a scenario giving a minimum response time in the loaded architecture.

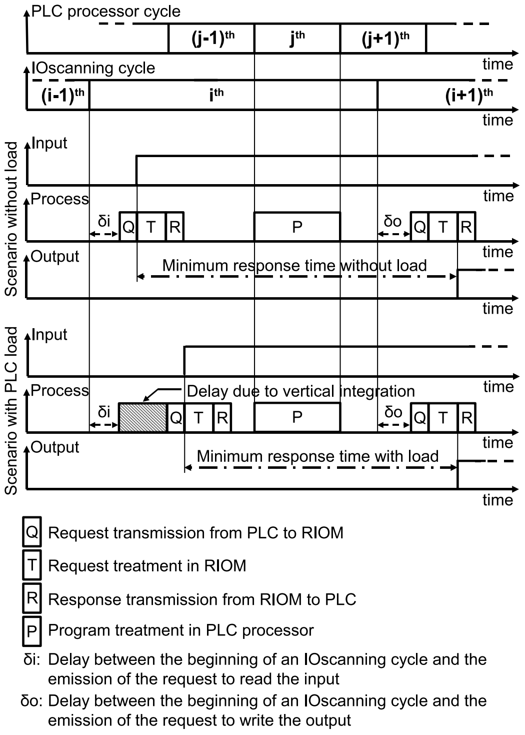
PLC processor cycle
(j-1)th
jth
(j+1)th
time
IOscanning cycle
(i-1)th
ith
(i+1)th
time
Scenario without load
Input
time
Process
δi
Q
T
R
P
δo
Q
T
R
time
Output
Minimum response time without load
time
Scenario with PLC load
Input
time
Process
Delay due to vertical integration
δi
Q
T
R
P
δo
Q
T
R
time
Output
Minimum response time with load
time
Q Request transmission from PLC to RIOM
T Request treatment in RIOM
R Response transmission from RIOM to PLC
P Program treatment in PLC processor
δi: Delay between the beginning of an IOscanning cycle and the emission of the request to read the input
δo: Delay between the beginning of an IOscanning cycle and the emission of the request to write the output

Figure 12. How can a loaded architecture provide shorter response times?

For the loadless scenario, when the i^{th} IOscanning cycle starts, the request to read RIOM R4 is transmitted – activity Q – after a necessary delay (δ_i) to emit requests to the previous RIOMs. The RIOM then treats the request – activity T – and the response is transmitted to the PLC – activity R. As soon as the next PLC processor cycle starts (the j^{th}, here), the reaction is computed – activity P. It is during the next – $(i+1)^{th}$ – IOscanning cycle that the computed reaction is sent to the RIOM.

The latest occurrence of the input event that can be taken into account is the one which occurs right before the beginning of the treatment activity T of the RIOM. The timing diagram of the input event for the loadless scenario in Fig. 12 illustrates this latest occurence. For the loaded scenario, after the beginning of the i^{th} IOscanning cycle, the sending of requests to the RIOMs can be delayed by a request coming from vertical integration – hatched rectangle. This delay can allow to take into account the occurrence of the input event later than with the loadless scenario. The timing diagram of the input event for the loaded scenario in Fig. 12 illustrates this later occurence. Thanks to this new delay, the minimum response time is smaller with vertical integration traffic.

6. Analysis

For each designed experiment presented in section 4, the current section provides an analysis of the results. Measurement results will be presented in a synthetic way by plots showing the evolutions of the value of the maximum response time according to a load parameter in X-coordinate. A horizontal asymptote at 22.25 ms is drawn on each graph. It corresponds to the reference response time, i.e. the maximum value of the response times measured without vertical integration load.

6.1. Impact of network sharing

For the impact study of the switch load, the selected parameters are the load frame size and the load throughput. The frame sizes that are taken into account are 100, 500 and 1500 bytes (maximal Ethernet frame size), and the load throughput is selected between 0 and 300 Mbit/s. The variation of the frame size allows to vary the frame throughput for a given bit throughput. For example a 100 Mbit/s throughput corresponds to 125 kiloframe/s for 100-bit-sized frames, and corresponds to 8.3 kiloframe/s for 1500-bit-sized frames. Whatever the frame size, the results are identical. Fig. 13 shows one of them. No significant impact is observable.

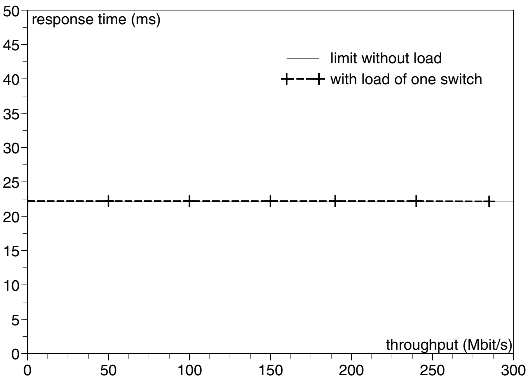

Figure 13. Response time with Ethernet switch load

For the impact study of the Ethernet link load, the parameters are the same but the maximum load throughput is limited to 200 Mbit/s – theoretical limit. Once again, whatever the frame size, the results are identical. As shown in Fig. 14, Ethernet links could be saturated. However up to 90% of the maximal throughput, there is no significant impact on response time.

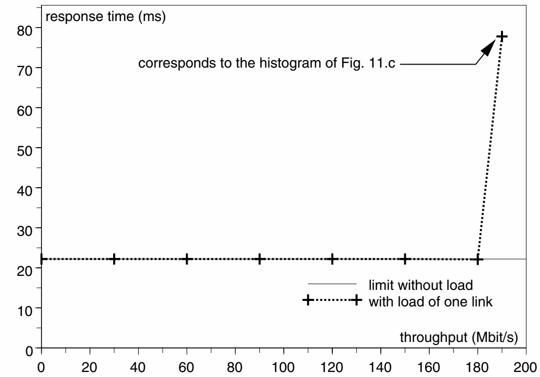

Figure 14. Response time with Ethernet link load

6.2. Impact of RIOM sharing

For impact study of RIOM load, the selected parameters are the number of clients for the 9 RIOMs and the request period for each client. The numbers of clients taken into account are 3, 6 and 9, and the period is selected between 3 and 50 ms. Measurement results are presented in Fig. 15.

Each plot includes two distinct parts. Above a threshold of request period, the RIOM load impact on response time is lower than 10% whereas below this period, an increase strongly appears. The threshold period is equal to 25 ms – respectively 10 ms and 5 ms – when 9 clients are requesting (respectively 6 clients and 3 clients). These good performances are probably due to the specialization of RIOMs. Their only function is to act as Modbus TCP/IP server.

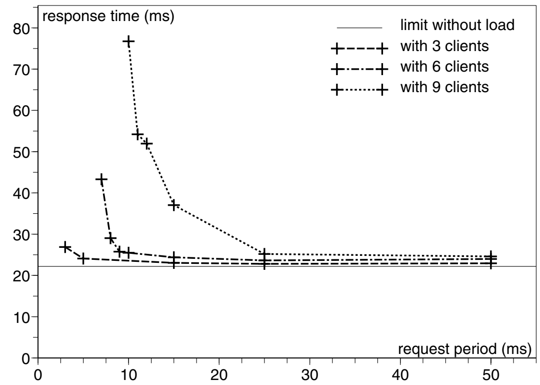

Figure 15. Response time with remote IO modules load

6.3. Impact of PLC sharing

For the impact study of the controller load, the parameters are the same than for the previous study, but the period range is larger. The numbers of clients taken into account are 3, 6 and 9, and the period is selected between 10 and 500 ms. Measurement results are presented in Fig. 16.

Even for the small loads corresponding to the periods above 100 ms, the impact is significant. The response time increase is greater than 45%. This can be explained by the versatility of the component. The controller carries out three functions simultaneously: process the control program, be the client of the RIOMs, and be server for the clients of vertical integration. With these conditions, it is less effective than the components specialized in only one function such as RIOMs.

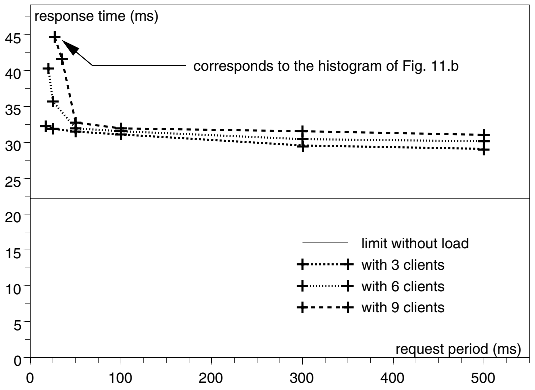

Figure 16. Response time with controller load

6.4. Feedback on vertical integration

From the previous results of the impact of elementary loads on response time, we will try to extrapolate the overall impact of vertical integration.

The first significant result is the absence of switch load impact. The network does not disturb response time as long as it is not saturated. Saturation is avoided below 90% of the theoretical throughput of one Ethernet port. Thus, with control functions, applications generating strong traffics can cohabit. For example, uploading control programs in stopped controllers, data logging or maintenance application traffic from wireless connection access on the field with a tablet PC.

To illustrate the previous results, two vertical integration configurations will be considered. They consist of three SCADA computers which scan their field equipments according to the same fast period (50 ms) and the same type of Modbus requests (read one register and write one register together). Moreover, these three computers scan the same equipments. For the configuration named $Config1$, these equipments are the two controllers $PLC1$ and $PLC2$, whereas for the configuration named $Config2$, they are the 9 RIOMs.

To use the synthetic plots of measurement results, it is necessary to determine the frame size and the generated throughput. A Read/Write register Modbus exchange generates an Ethernet frame of 89 bytes (19 for the application layer, 24 for the TCP layer, 20 for the IP layer and 26 for the Ethernet layers) for the request. An Ethernet frame of 81 bytes is generated for the response, thus 170 bytes (89+81) are flowing in the network every 50 ms for each client-server couple; the throughput is equal to 27.2 kbit/s. $Config1$ includes 3 client-server couples with $PLC1$ and 3 with $PLC2$ for vertical integration. The corresponding network load is $3 * 2 * 27.2 = 163.2$ kbit/s. $Config2$ includes 3 client-server couples for each RIOM. The corresponding network load is $3 * 9 * 27.2 = 734,4$ kbit/s. Table 1 gathers the values extracted from Fig. 13, 14, 15 and 16. The impact on the PLC for $Config1$ is obtained from plot "with 3 clients" in Fig. 16. A request period of 50 ms increases the response time by 22.25 to 32.00 ms, that is to say by 45%. $Config1$ does not request RIOMs, thus there is no impact.

Table 1. impact of two vertical integration configurations

	Switch impact	Eth port impact	PLC impact	RIOM impact	**Total impact**
Config1	0%	0%	45%	0%	**45%**
Config2	0%	0%	0%	<10%	**<10%**

The estimated total impact of each configuration shows that $Config1$, which generates the smaller traffic load (163.2 kbit/s), has a worse impact on response time. However it is this configuration which generally seems more natural to designers. In our case, it is preferable to limit the controller load rather than limit the network one. In the context of the vertical integration of the SCADA application, to get process data, designers must prefer to directly collect data from RIOMs rather than get them from the controllers.

7. Conclusions

Modbus TCP/IP with switched Ethernet fieldbus and client server protocol is generally used for non-real-time applications. This paper has shown that, with some care, it could be used for real time class 1 applications.

Thanks to its communication based on the TCP/IP stack and the fact that it is an open protocol, vertical integration is highly facilitated. To determine the impact of vertical integration on the response times in a control architecture, we have defined which kinds of load appear on the different components. For the different loads, we have created load generators to excite the components.

A very high number of measures has been made to quantify the different impacts. Two main results were highlighted. First and foremost, it was shown that the load in the network components – switches, Ethernet links – has no impact at all on the response time. The limiting load of the network is of 90%, which is far higher than generally expected. The study of load scenarios on automation components – controllers and RIOMs – showed that, in order to get information from the process without disturbing the control, it is preferable to query the RIOMs rather than the controllers. Again, this is a very interesting result since the current standard approach consists of querying the controller.

References

[1] P. Ferrari, A. Flammini, and S. Vitturi. Performance analysis of profinet networks. *Computer Standards and Interfaces*, 28:369–385, 2006.

[2] J.-P. Georges, E. Rondeau, and T. Divoux. How to be sure that Ethernet networks will satisfy the real-time requirements? In *Proc. of IEEE Int. Symposium on Industrial Electronics*, July 2002.

[3] J. Greifeneder and 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.

[4] W. B. Hatfield, F. W. Scholl, M. H. Coden, R. W. Kilgore, and J. H. Helbers. Fiber optic Ethernet in the factory automation environment. In *Proc. of SPIE Conference on Fiber Optic Networks and Coherent Technology in Fiber Optic Systems*, pages 297–302, 1987.

[5] M.-H. Hung, J. Tsai, F.-T. Cheng, and H.-C. Yang. Development of an ethernet-based equipment integration framework for factory automation. *Robotics and Computer-Integrated Manufacturing*, 20:369–383, 2004.

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

[7] G. Marsal. *Evaluation of time performances of Ethernet-based automation systems by simulation of high-level Petri nets*. PhD thesis, ENS Cachan (France) and Kaiserslautern University (Germany), december 2006.

[8] P. Neumann. Communication in industrial automation - what is going on? *Control Engineering Practice*, doi:10.1016/j.conengprac.2006.10.004, 2006.

[9] N. Pereira, E. Tovar, and L. M. Pinho. Timeliness in COTS factory-floor distributed systems: what role for simulation? In *Proc. of 9th IEEE Int. Conf. on Emerging Technologies and Factory Automation*, Sept. 2004.

[10] H. Schutz. The role of MAP in factory integration. *IEEE Transactions on Industrial Electronics*, 35(1):6–12, 1988.

[11] R. Viegas, R. Valentim, D. Texeira, and L. Guedes. Analysis of protocols to ethernet automation networks. In *Proc. of International Joint Conference SICE-ICASE*, pages 4981–4985, 2006.

[12] S. Vitturi. On the use of Ethernet at low level of factory communication systems. *Computer Standards and Interfaces*, 23:267–277, 2001.

[13] 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 (INCOM 2006)*, pages 101–106, 2006.

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