

Evaluating the Performance of Fault-Tolerant S2A vs. In-Loop Controller Models for Ethernet-Based NCS

Eslam Moustafa, Hassan Halawa, Ramez Daoud, Hassanein Amer

Electronics Engineering Department, American University in Cairo, Cairo, Egypt
Email: eslam_14@aucegypt.edu, hassan@aucegypt.edu, rdaoud@ieee.org, hamer@aucegypt.edu

Received 22 March 2014; revised 22 April 2014; accepted 29 April 2014

Copyright © 2014 by authors and Scientific Research Publishing Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY).
<http://creativecommons.org/licenses/by/4.0/>



Open Access

Abstract

Two different controller-level fault-tolerant models for Ethernet-based Networked Control Systems (NCSs) are presented in this paper. These models are studied using unmodified Fast and Gigabit Ethernet. The first is an in-loop controller model while the second is a direct Sensor to Actuator (S2A) model. OMNeT++ simulations showed the success of both models in meeting system end-to-end delay and strict zero packet loss (with no over-delayed packets) requirements. It was shown in the literature that the S2A model has a lower end-to-end delay than the in-loop controller model. It will be shown here that, on the one hand, the in-loop fault-tolerant model performs better in terms of less total end-to-end delay than the S2A model in the fault-free situation. While, on the other hand, in the scenario with the failed controller(s), the S2A model was shown to have less total end-to-end delay.

Keywords

Networked Control Systems, Ethernet, OMNeT++, Sensor Actuator Networks, Fault-Tolerance

1. Introduction

Manufacturing control has been moving more and more towards distributed implementations of control systems. Networks are used to communicate the data instead of using traditional point-to-point communication. Networks require less wiring and less maintenance compared to a point-to-point architecture. Such networks carry a large number of small control signals between many nodes and these signals have to meet the delay constraints of real-time control systems. The main difference between such control networks and conventional data networks is that control networks must be able to support time-critical applications [1].

Networked Control Systems (NCSs) consist of sensors, controllers and actuators that communicate together over a network. Sensors send packets to the controllers which calculate the control action that should be delivered to the actuators, and these transmissions must meet the control system's deadline. There are four factors that affect the utilization of the network bandwidth: the sampling rate, the number of nodes requiring synchronous operation, the size of the information sent and the protocol used. Traditionally, for proper control, there are different protocols used which have a deterministic behavior such as DeviceNET and ControlNET [2] [3]. Also, many real-time applications were studied using protocols such as Controller Area Network (CAN), PROFIBUS and EtherNet/IP which is a merger between Ethernet and ControlNET [3]-[7].

Recently, Ethernet has been used in Networked Control Systems. Although Ethernet is a nondeterministic protocol, it was shown, in many studies, that it can be used in NCSs [1] [8]-[19]. Ethernet is widespread nowadays in communication systems because it has been proven to be a very successful protocol. By using such a protocol, installation and maintenance costs can be reduced in industrial applications. In [10], real-time as well as non-real time traffic were integrated without any changes to the IEEE 802.3 protocol. It was shown that a system consisting of 16 sensors, one controller and 4 actuators was able to meet the required time constraints. The system was running on-top-of Gigabit Ethernet. On the other hand, in [20], it was shown that an architecture with a integrated control design was also a possible solution where each actuator would have its own controller integrated in the same node instead of one controller for the entire system as in [10]. This same concept is implemented in Sensor Actuator Networks (SANETs) where a group of sensors and actuators are distributed geographically and communicate together through wired or wireless networks [21]-[23].

Different fault-tolerance techniques were applied previously on the node level in Networked Control Systems. A supervisory control level is essential in many distributed control systems where the functions are hierarchal. The role of this level is monitoring the control objectives and supporting the overall coordinated control in different phases of normal operation. Also, this level allows the diagnosis of all foreseeable faults, takes the necessary corrective actions, including the change of controller parameter or structure [24]. In [25], a hierarchical control architecture was discussed where a supervisor operated on top of separate controller nodes. It was shown via simulations that the system was fully functional. Fault-tolerance was added to these aforementioned architectures and reliability/availability models were used to quantitatively assess the increase in lifetime [26] [27]. In [28], actuator fault-tolerant architecture was presented in order to detect all relevant faults of an electrical steering system by using a double stator AC motor instead of duplicated motors. The paper showed how active control reconfiguration can accommodate all critical faults which were demonstrated on the hardware of a warehouse truck. There are other ways of analyzing the fault-tolerant problem for the networked control systems (NCSs) such as using fuzzy models [29]. A Takagi-Sugeno (T-S) fuzzy model with parametrical uncertainties was used to approximate the T-S model where robust controllers were designed with sensors or actuators failure. It was shown via simulations that the method is effective and the system can be kept asymptotically stable under some sensors failures or actuators failures.

The focus in this research is the development of new architectures where fault-tolerance techniques are applied and studied on the controller level using unmodified Fast Ethernet and Gigabit Ethernet as they have sufficient bandwidth to easily handle the data volume. Two architectures will be studied; the first is an in-loop controller architecture while the second is a direct Sensor to Actuator (S2A) architecture. Additionally, a performance comparison in terms of the total end-to-end delay between the two architectures, will be carried out in the fault-free scenario and the scenario where there are failed controller(s).

The rest of the paper is organized as follows. Section 2 presents more details about related work. Section 3 elaborates on the new proposed models. In Section 4, the simulation results are presented and discussed. Finally, the paper is concluded in Section 5.

2. Previous Work

In [20], NCSs were studied where actuators would have their own control function; it was shown that this integrated approach to the design of real-time fieldbus-based distributed control systems was successful in meeting control requirements. At first, the performance of the system suffered severe degradation, but architecture and control co-design were used to achieve a successful implementation of this distributed system. In [10], real-time and non-real time traffic were integrated in an NCS without any modifications to the IEEE 802.3 protocol packet format. It was shown in a mixed traffic industrial environment that standard Gigabit Ethernet succeeds in meet-

ing timing constraints. In [30], a new model was developed; it has 16 sensors and 4 actuators as in [10]. However, each sensor communicates with the appropriate actuator(s) directly without going through a controller node. In other words, each actuator incorporates its own control function as in [20]. This proposed system was studied on-top-of both Fast and Gigabit switched Ethernet. It was shown that this architecture succeeds in meeting the required time constraints. Then, the architecture was compared to the architecture in [10] and it was shown, via OMNeT++ [31] simulations, that the observed end-to-end delay is smaller in the proposed architecture. Finally, it was shown that the proposed model can withstand more additional load than a system with an in-loop controller as in [10].

In this paper, new models are developed where the focus is on applying fault-tolerance techniques on the control level of both architectures in [10] and [30]. These fault-tolerance techniques will increase the reliability of the architectures as well as their lifetime. New models are developed on-top-of both Fast and Gigabit switched Ethernet. Comparison is made between the two proposed models via OMNeT++ simulations where the focus is on factors such as the number of packets dropped and the observed end-to-end delay.

3. Proposed Models

In this section, a comparison is made between two different control network fault-tolerance models. The in-loop fault-tolerant model is based on the one in [10] while the S2A one is based on [30]. In the in-loop model, there are two controllers which receive the packets from the sensors and only one of them sends control packets to the actuators while the other one is in hot-standby mode as shown in **Figure 1**. In the S2A model, a supervisor is responsible for monitoring network behavior by receiving packets from all the different nodes in the network as shown in **Figure 2**. The controlling process in the in-loop model takes place in an individual controller node, while in the S2A model it takes place in the smart actuator node(s) which are more intelligent nodes where both the control and actuation processes occur. To incorporate fault-tolerance into the S2A model, two controllers will be used per actuator where both controllers receive packets from the sensors but only one of them is chosen, via a multiplexer, to send the control packets to the actuators as shown in **Figure 3**. Both models are operated under Fast or Gigabit Ethernet links where the focus is on the packets dropped and the observed end-to-end delay. Note that the end-to-end delay includes all types of encapsulation/decapsulation, propagation and queuing delays.

3.1. Models Description

The in-loop fault-tolerant model is based on the one used in [10] as shown in **Figure 1**. It consists of 16 sensors, two controllers and 4 actuators. All 16 sensors send their data to both controllers where one of them is active and the other one is in hot-standby mode. The active controller then computes the control action and transmits it to the 4 actuators. Both controllers send watchdog signals to each other and if the active controller fails, the hot-standby controller will be alerted via the absence of the watchdog signal; therefore, it would take over and become the active controller.

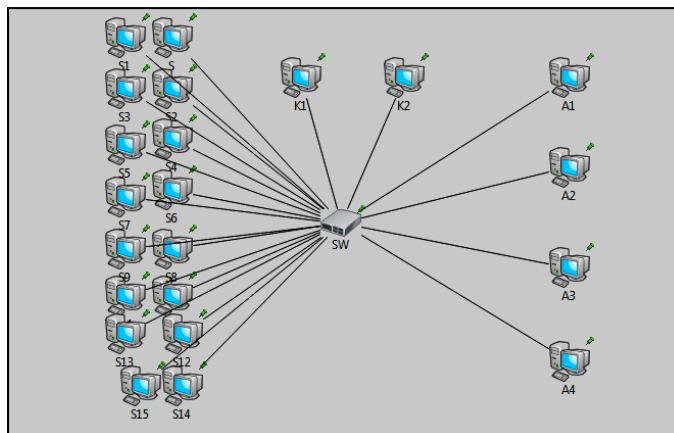


Figure 1. Fault-tolerant in-loop model architecture.

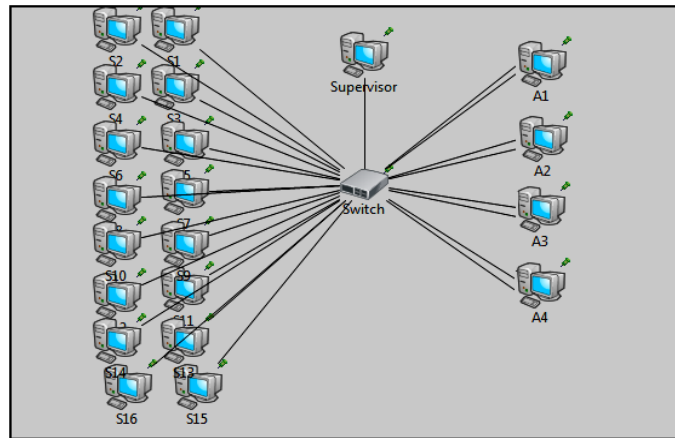


Figure 2. Fault-tolerant S2A model architecture.

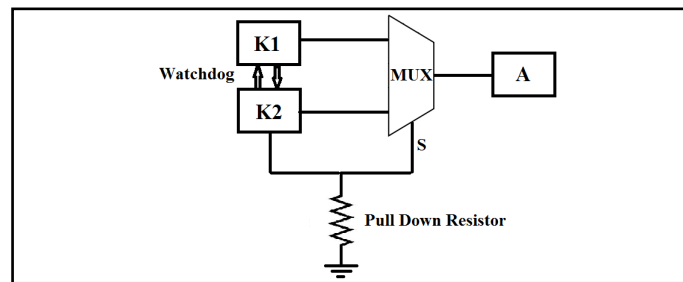


Figure 3. Fault detection and recovery mechanism (S2A model).

The S2A model also consists of 16 sensors which send data directly to the 4 actuators but instead of a controller, a supervisor is used; all sensors and actuators send packets to the supervisor node which is responsible for monitoring the behavior of the network. The S2A fault-tolerance model is based on the one used in [30] as shown in **Figure 2**. Each one of the actuators will have two controllers integrated in the same node. The two controllers receive data from the 16 sensors and send watchdog signals to each other. They are connected to the actuator (A) via a multiplexer integrated in the same node on a circuit board as shown in **Figure 3**. In a fault-free scenario, the first controller (K1 in **Figure 1**) is working and sending data to the actuator while the second controller is inactive by sending “0” to the selection line (S) of the MUX in order not to be chosen. Upon the failure of the first controller, the second one will be alerted via the absence of the watchdog signal; therefore it would send a “1” to the selection line of the MUX to be chosen and becomes the active one sending the data to the actuator. Furthermore, the assumed fault model for the second controller is an open circuit output. By using a pull down resistor, “0” will be sent to the MUX selection line to keep the active controller connected to the output of the MUX and continue receiving the data from the first one as shown in **Figure 3**. Note that the watchdog signal in the in-loop model is sent over the network while the watchdog signal in the S2A model is sent on the circuit level.

OMNeT++ is used as the simulation platform. All the nodes including sensors, controllers, supervisor, and actuators are modeled using standard hosts. Control packets are communicated on-top-of UDP [32]. Also, the payload is fixed at 100Bytes. The sampling frequency used in the two models is 1440 Hz based on a 1440 electric pulses encoder for 360 degrees shaft rotation assuming one revolution per second [33]. Therefore, the control action must be taken within a time frame of 694 μ s. Watchdog signals are sent over the network in the in-loop model every 347 μ s which is half of the sampling period in order not to lose any samples upon the failure of the active controller. Finally, both models are compared once on-top-of Fast Ethernet and again on-top-of Gigabit Ethernet.

3.2. Analysis

This subsection presents an analysis to calculate the end-to-end delay for both fault-tolerant models mentioned

above using both Fast and Gigabit switched Ethernet. A worst-case delay analysis will be carried out on both models. The presented analysis aims to model, calculate and contrast the end-to-end delays resulting from the periodic nature of the control traffic in both models. For this analysis, the focus will be on the last packet being transmitted by the final sensor node. The flow of this packet represents the worst-case scenario as all previously sent packets are queued up ahead of it as shown in **Figure 4**.

For both models, without accounting for processing delays [34], the amount of time required for the transmission of a single packet over a particular link is given by:

$$D_{\text{packet}} = D_{\text{transmission}} + D_{\text{propagation}} \quad (1)$$

The Link Transmission delay ($D_{\text{transmission}}$) is the amount of time required for all of the packet's bits to be transmitted onto the link and it is a function of the packet length L (bits) and link transmission rate R (bps) [35].

$$D_{\text{transmission}} = L/R \quad (2)$$

The length of the packet is fixed at 100 Bytes at the application layer; however, additional packet and frame header overhead (approximately 58 Bytes) must be taken into consideration. All the links are Gigabit Ethernet in one scenario and Fast Ethernet in the second scenario, therefore

$$\text{Gigabit Ethernet: } D_{\text{transmission}} = (158 \times 8) / (10^9) = 1.264 \mu\text{s} \quad (3)$$

$$\text{Fast Ethernet: } D_{\text{transmission}} = (158 \times 8) / (10^8) = 12.64 \mu\text{s} \quad (4)$$

The propagation delay ($D_{\text{propagation}}$) is the time taken for the packet to travel from the sender to the receiver; it is a function of the link length d (m) and the propagation speed s (m/s) [35].

$$D_{\text{propagation}} = d/s \quad (5)$$

The length between each node and the switch is $d = 1.5$ m and the transmission speed in the Ethernet links is $s = 2 \times 10^8$ m/s.

$$D_{\text{propagation}} = 1.5 / (2 \times 10^8) = 0.0075 \mu\text{s} \quad (6)$$

The total end-to-end delay for the worst-case packet flow is given by:

$$D_{\text{total}} = \text{Total Number of Packets Transmitted Sequentially} \times D_{\text{packet}} \quad (7)$$

In the Fault-Free Scenario, the number of packets is 23 in the in-loop model and 24 in the S2A model as shown in **Figure 4**, therefore the total delay can be calculated using Equation (7) as shown below:

In-Loop Model

$$\text{Fast Ethernet: } D_{\text{total}} = (23 \times (12.64 + 0.0075) \times 10^{-6}) = 290.893 \mu\text{s} \quad (8)$$

$$\text{Gigabit Ethernet: } D_{\text{total}} = (23 \times (1.264 + 0.0075) \times 10^{-6}) = 29.245 \mu\text{s} \quad (9)$$

S2A Model

$$\text{Fast Ethernet: } D_{\text{total}} = (24 \times (12.64 + 0.0075) \times 10^{-6}) = 303.540 \mu\text{s} \quad (10)$$

$$\text{Gigabit Ethernet: } D_{\text{total}} = (24 \times (1.264 + 0.0075) \times 10^{-6}) = 30.516 \mu\text{s} \quad (11)$$

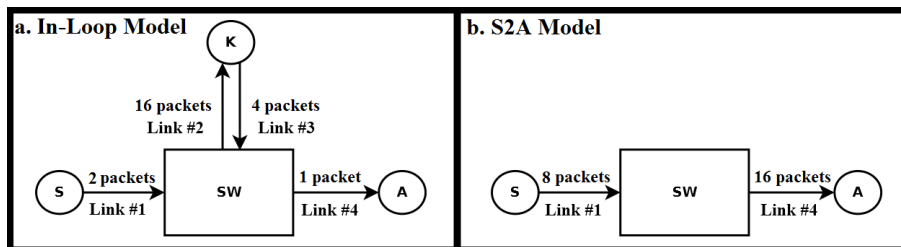


Figure 4. Worst-case packet flow analysis for the fault-free scenario.

On the other hand, upon the failure of one of the controllers in the in-loop model and the failure of one controller per actuator in the S2A model, the worst case packet flow analysis will be 22 for the in-loop model and 20 for the S2A model as in [30]. Therefore, the D_{total} calculations can be calculated using Equation (7) as shown below:

In-Loop Model

$$\text{Fast Ethernet : } D_{total} = (22 \times (12.64 + 0.0075) \times 10^{-6}) = 278.245 \mu\text{s} \quad (12)$$

$$\text{Gigabit Ethernet : } D_{total} = (22 \times (1.264 + 0.0075) \times 10^{-6}) = 27.973 \mu\text{s} \quad (13)$$

S2A Model

$$\text{Fast Ethernet : } D_{total} = (20 \times (12.64 + 0.0075) \times 10^{-6}) = 252.95 \mu\text{s} \quad (14)$$

$$\text{Gigabit Ethernet : } D_{total} = (20 \times (1.264 + 0.0075) \times 10^{-6}) = 25.43 \mu\text{s} \quad (15)$$

A summary of the theoretical results for both in-loop and S2A models in fault-free and failed controller(s) scenarios is shown in **Table 1**.

4. Simulation Results

In this section, OMNET++ simulation results are presented. In all simulations, there were no packets dropped.

4.1. Fault-Free Scenario

OMNeT++ simulations are carried out for both fault-tolerant models: the in-loop based on the model in [10] and the S2A model based on the one in [30]. Using Fast Ethernet, the in-loop model had a smaller maximum end-to-end delay of 305.470 μs compared to 318.734 μs for the S2A model. Note that the 305.470 μs delay is the sum of the 239.055 μs (maximum sensor to controller end-to-end delay) and 66.415 μs (maximum controller to actuator end-to-end delay) for the in-loop model as shown in **Figure 5** and **Figure 6** where the data travels over two hops. Also, the 318.734 μs delay is that for the actuator node only in the S2A model as shown in **Figure 7** where the data travels over one hop. Similarly, using Gigabit Ethernet, the maximum end-to-end delay was found to be 30.574 μs for the in-loop model and 31.886 μs for the proposed model. This means that the in-loop model performs better, which is expected, due to the fact that there are two separate controllers in the in-loop model while there are 8 controllers (2 controllers per actuator) in the S2A thus increasing the amount of traffic in the network which increases the experienced end-to-end delay. In the Figures, the x-axis represents the Simulation Time (seconds) and the y-axis shows the End-to-end Delay (seconds).

4.2. Scenario with the Failed Controller(s)

On the other hand, upon the failure of one of the two separate controllers in the in-loop model or one controller from each of the four pairs of integrated controllers in the S2A model, it was found that the S2A model performs better with less end-to-end delay. This is due to the fact that traffic sent in the in-loop model must go through additional intermediate hops via the controller. In the S2A model, only one hop is needed to transmit the traffic thus decreasing the experienced end-to-end delay. In the scenario with the failed controller(s), using Fast Ethernet, the S2A model had a smaller maximum end-to-end delay of 265.615 μs compared to 292.189 μs for the in-loop model. Note that the 292.189 μs delay is the sum of the 225.77 μs (maximum sensor to controller end-to-

Table 1. Worst-case end-to-end delay analysis results summary (in μ seconds).

Scenario	Link Speed	Fault-Free Theoretical Result	Failed Controller(s) Theoretical Result
In-Loop Model	100 Mbps	290.893	278.245
	1 Gbps	29.245	27.973
S2A Model	100 Mbps	303.540	252.95
	1 Gbps	30.516	25.43

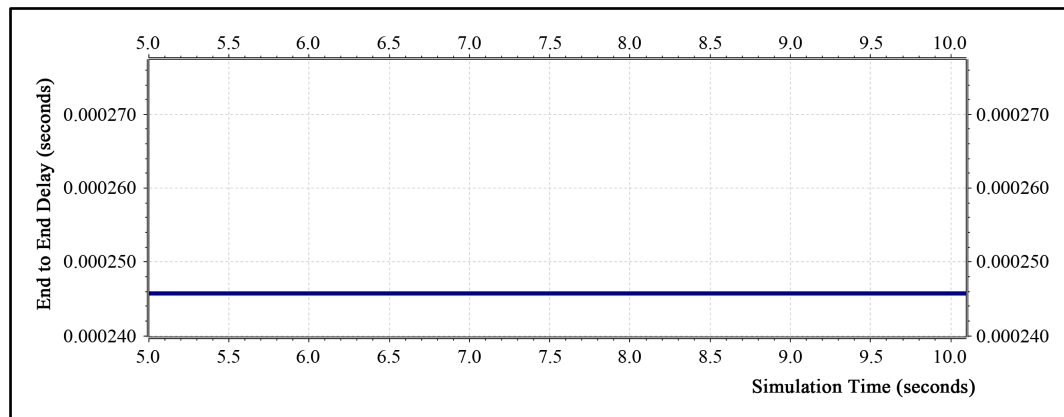


Figure 5. Constant maximum end-to-end delay between the 16 sensors and the controller (In-Loop Model).

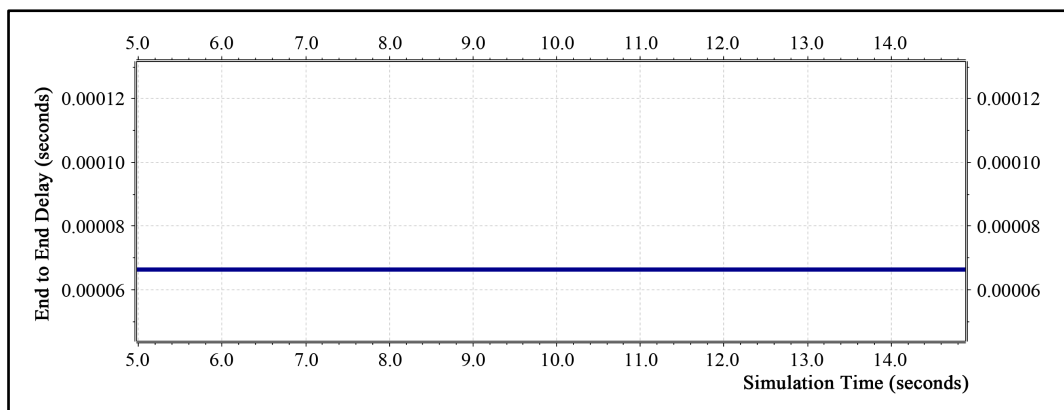


Figure 6. Constant maximum end-to-end delay between the controller and the 4 actuators (In-Loop Model).

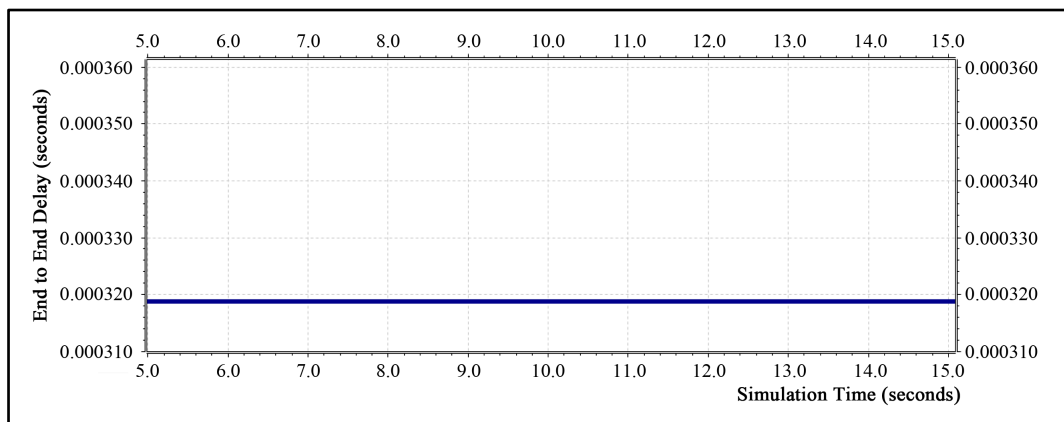


Figure 7. Constant maximum end-to-end delay between the 16 sensors and the 4 actuators (S2A Model).

end delay) and $66.419 \mu\text{s}$ (maximum controller to actuator end-to-end delay) for the in-loop model. Similarly, using Gigabit Ethernet, the maximum end-to-end delay was found to be $29.245 \mu\text{s}$ for the in-loop model and $26.575 \mu\text{s}$ for the S2A model. In conclusion, using both Fast and Gigabit Ethernet, the in-loop model showed less delay in the fault-free scenario while the S2A model showed less delay in the scenario with the failed controller(s) as summarized in [Table 2](#) and [Table 3](#). Also, [Table 2](#) and [Table 3](#) show the percentage error between calculated and simulated results. Note that, due to the regularity of the traffic imposed on the network, packets experience the same end-to-end delay for each sample.

Table 2. Theoretical and simulation results in fault-free scenario (in μ seconds).

Scenario	Link Speed	Theoretical Result	Simulation Result	Error %
In-Loop Model	100 Mbps	290.893	305.470	4.77%
	1 Gbps	29.245	30.574	4.34%
S2A Model	100 Mbps	303.540	318.734	4.76%
	1 Gbps	30.516	31.886	4.29%

Table 3. Theoretical and simulation results in scenario with the failed controller(s) (in μ seconds).

Scenario	Link Speed	Theoretical Result	Simulation Result	Error %
In-Loop Model	100 Mbps	278.245	292.189	4.77%
	1 Gbps	27.973	29.245	4.35%
S2A Model	100 Mbps	252.95	265.615	4.77%
	1 Gbps	25.43	26.575	4.31%

5. Conclusions

A major system design requirement for Networked Control Systems (NCSs) is to meet real-time delay requirement for sensors, controllers and actuators. Different protocols were studied to maintain requirements of speed and correctness such as Controller Area Network (CAN) and PROFIBUS. Ethernet in NCS is now one of the most widespread and low cost protocols available. Fault-tolerance techniques are required to increase reliability and lifetime of such networks. Therefore, in this paper, new models were studied on-top-of both Fast and Gigabit switched Ethernet. Comparison was made between the two proposed models via OMNeT++ simulations where the focus is on factors such as the packets dropped and the observed end-to-end delay. The delay measurement includes all types of encapsulation/decapsulation, propagation and queuing delays.

It was shown in the literature that the S2A model has a lower end-to-end delay than an in-loop controller model. In this paper, it was shown that both of the proposed architectures were successful in meeting all required timing constraints and no packets were dropped. However, the in-loop model performs better in terms of less total end-to-end delay than the S2A model in the fault-free situation because there are two separate controllers compared with 8 controllers (two controllers per actuator) in the S2A model thus increasing the delay. On the other hand, in the scenario with the failed controller(s), the S2A model was shown to have less total end-to-end delay. The traffic in the direct S2A model is sent over one hop compared to the in-loop model where the traffic must go through an additional intermediate hop via the controller thus increasing the experienced end-to-end delay. Different scenarios were tested including different link bandwidths (Fast or Gigabit Ethernet) in both the fault-free scenario and the scenario with the failed controller(s).

References

- [1] Lian, F.L., Moyné, J.R. and Tilbury, D.M. (2001) Performance Evaluation of Control Networks: Ethernet, ControlNet, and DeviceNet. *IEEE Control Systems Magazine*, **21**, 66-83. <http://dx.doi.org/10.1109/37.898793>
- [2] ODVA, Volume 1: CIP Common. http://www.odva.org/10_2/03_events/03_ethernet-homepage.htm
- [3] ODVA, Volume 2: EtherNet/IP Adaptation on CIP. http://www.odva.org/10_2/03_events/03_ethernet-homepage.htm
- [4] Official Site for PROFIBUS and PROFINET. <http://www.profibus.com>
- [5] Bosch (1991) CAN Specification version 2.0 ISO 11898.
- [6] EtherNet/IP Performance and Application Guide, Allen-Bradley, Rockwell Automation, Application Solution.
- [7] Official Site for Control Net. <http://odva.org/default.aspx?tabid=244>
- [8] Nilsson, J. (1998) Real-Time Control Systems with Delays, Ph.D. Thesis, Department of Automatic Control, Lund Institute of Technology, Lund.
- [9] Lian, F.L., Moyné, J.R. and Tilbury, D.M. (2001) Networked Control Systems Toolkit: A Simulation Package for

Analysis and Design of Control Systems with Network Communication. Technical Rep., UM-ME-01-04, July 2001.

<http://www.eecs.umich.edu/~impact>

- [10] Daoud, R., Elsayed, H., Amer, H. and Eid, S. (2003) Performance of Fast and Gigabit Ethernet in Networked Control Systems. *Proceedings of the 46th IEEE Midwest Symposium on Circuits and Systems MWSCAS*, Cairo, December 2003. <http://dx.doi.org/10.1109/MWSCAS.2003.1562328>
- [11] Tolly, K. (1997) The Great Networking Correction: Frames Reaffirmed. Industry Report, The Tolly Group, IEEE Internet Computing.
- [12] Wittenmark, B., Bastian, B. and Nilsson, J. (1998) Analysis of Time Delays in Synchronous and Asynchronous Control Loops. *Proceedings of the 37th IEEE Conference on Decision and Control*, Tampa, December 1998, 283-288.
- [13] Wang, J. and Keshav, S. (1999) Efficient and Accurate Ethernet Simulation. Cornell Network Research Group (C/NRG), Department of Computer Science, Cornell University.
- [14] Moss, B. (2000) Real-time Control on Ethernet. *Dedicated Systems*, No. 00q2, 53-60.
- [15] EtherNet/IP Performance and Application Guide. Allen-Bradley, Rockwell Automation, Application Solution.
- [16] Lounsbury, B. and Westerman, J. (2001) Ethernet: Surviving the Manufacturing and Industrial Environment. Allen-Bradley White Paper.
- [17] Lee, S.H. and Cho, K.H. (2001) Congestion Control of High-Speed Gigabit-Ethernet Networks for Industrial Applications. *Proceedings of IEEE International Symposium on Industrial Electronics*, Pusan, 12-16 June 2001, 260-265.
- [18] Daoud, R.M., Amer, H.H., Elsayed, H.M. and Sallez, Y. (2006) Ethernet-Based Car Control Network. *Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering CCECE*, Ottawa, 7-10 May 2006.
- [19] IEC 61784-1 and 61784-2. www.iec.ch
- [20] Marti, P., Fuertes, J.M. and Fohler, G. (2001) An Integrated Approach to Real-Time Distributed Control Systems over Fieldbuses. *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Antipies-Juan les Pins, 15-18 October 2001, 177-182.
- [21] Zinonos, Z., Silva, R., Vassiliou, V. and Silva, J.S. (2011) Mobility Solutions for Wireless Sensor and Actuator Networks with Performance Guarantees. *Proceedings of the International Conference on Telecommunications ICT*, Ayia Napa, 8-11 May 2011, 406-411.
- [22] Cecilio, J., Martins, P., Costa, J. and Furtado, P. (2012) A Configurable Middleware for Processing in Heterogeneous Industrial Intelligent Sensors. *Proceedings of the IEEE International Conference on Intelligent Engineering Systems INES*, Lisbon, 13-15 June 2012, 145-150.
- [23] Aoki, S., Kirihara, Y., Nakazawa, J., Takashio, H. and Tokuda, H. (2009) A Sensor Actuator Network Architecture with Control Rules. *Proceedings of the 6th International Conference on Networked Sensing Systems INSS*, Pittsburgh, 17-19 June 2009, 1-4.
- [24] Blanke, M., Staroswiecki, M. and Wu, N. (2001) Concepts and Methods in Fault-Tolerant Control. *Proceedings of the American Control Conference*, Arlington, 25-27 June 2001, 2608-2620.
- [25] Daoud, R.M., Amer, H.H. and ElSayed, H.M. (2005) Fault Tolerant Two-Level Pyramid Networked Control Systems. *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Catania, 19-22 September 2005, CF 000471, 6 p.
- [26] Amer, H.H., Moustafa, M.S. and Daoud, R.M. (2006) Availability of Pyramid Industrial Networks. *Proceedings of the Canadian Conference on Electrical and Computer Engineering CCECE*, Ottawa, 7-10 May 2006, 1862-1865.
- [27] Amer, H.H. and Daoud, R.M. (2006) Parameter Determination for the Markov Modeling of Two-Machine Production Lines. *Proceedings of the International IEEE Conference on Industrial Informatics INDIN*, Singapore, 16-18 August 2006, 1178-1182.
- [28] Thomsen, J.S. and Blanke, M. (2006) Fault-Tolerant Actuator System for Electrical Steering of Vehicles. *Proceedings of the 32nd Annual Conference of the IEEE Industrial Electronics IECON*, Paris, 6-10 November 2006, 3597-3602.
- [29] Zhang, J.J. (2012) Fuzzy Robust Fault-Tolerant Control for Networked Control Systems. *Proceedings of the International Conference on Computer Science & Education ICCSE*, Melbourne, 14-17 July 2012, 1267-1270.
- [30] Moustafa, E.A., Halawa, H.H., Daoud, R.M. and Amer, H.H. (2013) Sensor Actuator Ethernet-Based Networked Control Systems. *Proceedings of the 14th International IEEE Conference on Sciences and Techniques of Automatic Control and Computer Engineering STA*, Sousse, 20-22 December 2013, 530-534.
- [31] Official Site for OMNeT++. www.omnetpp.org/
- [32] Boggia, G., Camarda, P., Divittorio, V. and Grieco, L.A. (2009) A Simulation-Based Performance Evaluation of Wireless Networked Control Systems. *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Mallorca, 22-25 September 2009, 1-6.

- [33] Official Site for SIEMENS. <http://www.siemens.com/entry/cc/en/>
- [34] Seno, L., Vitturi, S. and Tramarin, F. (2011) Experimental Evaluation of the Service Time for Industrial Hybrid (Wired/Wireless) Networks under Non-Ideal Environmental Conditions. *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation ETFA*, Toulouse, 5-9 September 2011, 1-8.
- [35] Kurose, J.F. and Ross, K.W. (2000) *Computer Networking: A Top-Down Approach Featuring the Internet*. Addison Wesley Publishing Company, Boston.