Re-evaluating Event-Triggered and Time-Triggered Systems

Jason J. Scarlett
University of Calgary
2500 University Dr. N.W.
Calgary, Alberta, CANADA, T2N 1N4
jason.scarlett@ucalgary.ca

Robert W. Brennan
University of Calgary
2500 University Dr. N.W.
Calgary, Alberta, CANADA, T2N 1N4
rbrennan@ucalgary.ca

Abstract

Industrial control system design is increasingly adopting Object Oriented (OO) approaches. These approaches use an event-triggered communication model where communication is triggered by the occurrence of a significant event. Traditional low-level control systems for safety-critical applications use a different approach, namely time-triggered.

The truth is that both event-triggered and time-triggered protocols have their advantages. Time-triggered protocols are dependable and event-triggered protocols are more responsive. Safety-critical systems demand dependability and as a result, time-triggered protocols have been used to date.

Promising new findings indicate that an event-triggered protocol can be both responsive and dependable: a first in safety-critical control systems.

This paper contrasts the two existing protocol paradigms and evaluation methodologies. A new protocol and new evaluation methodology are suggested that meet the stringent requirements of a safety-critical system. Initial findings are reported that indicate this to be a promising new direction.

1. INTRODUCTION

Control systems used in safety-critical applications are increasingly becoming distributed. This is driven by the increasing performance and cost efficiency of smart devices interconnected by fieldbuses (bi-directional digital serial networks). As well, this is motivated by the need to match the control model more closely with the physical system. This is particularly relevant to manufacturing control systems that are required to control widely distributed devices in an environment that is prone to disruptions. With this model, control is achieved by the emergent behaviour of many simple, autonomous and co-operative entities (i.e., agents) that "decide locally not only how to act (as subroutines do), and what actions to take (as objects do), but also when to initiate their own activity" [1]. The natural fit of multi-agent systems technology to manufacturing problems has resulted in many applications in this domain [2]. Consequently, this has led to Holonic Manufacturing Systems (HMS), a manufacturing-specific application of the broader MAS approach [3].

Recently, there has been considerable interest in extending this work from the upper, planning and scheduling level of control, to the physical device level. For example, members of the HMS consortium focused on defining a low-level control architecture [4, 5] that is based on the IEC 61499 function block standard. With this work as a basis, IEC 61499 function blocks have been used at the device level for dynamic reconfiguration [4, 6], safety management [7, 8], and system validation [9]. This has led to range of industrial applications of real time distributed control such as DaimlerChrysler's holonic control implementation for engine assembly [10], Rockwell Automation's design of a reconfigurable ship-based chilled-water system, the integration of RFID (radio frequency identification) systems [11], and our recent work with C-Core and DRDC (Defence Research and Development Canada) on the use of holonic systems for sensor management on military platforms [12].

When agent technology is applied to physical devices, system design and analysis are of utmost concern given real time and safety issues at this level of control. For example, as devices such as controllers, sensors and actuators become "smarter", safety functions that were previously performed by mechanical or electrical interlocks are assumed by computer software that may reside in a single device or might be distributed across multiple devices. Failures in these systems can have a significant impact on a company's bottom line (e.g., profit reduction due to temporary loss of equipment resulting in production downtime), and more importantly, on the lives of people. Because of the more stringent requirements for latency, reliability and availability, it follows that the step from the non-real time or soft real time domain is a large one, requiring new models and methodologies for distributed control.

As with previous architectures, a distributed architecture must deal with safety issues such as redundancy, data validation, fault isolation, and timing issues. Timing issues in a distributed system must consider how the network resources are managed

1-4244-0681-1/06/$20.00 ©2006 IEEE

because the network is now comprised of a single digital communication bus shared between all devices. Timing of communication must deal with unpredictable collisions that result in delays.

Advanced distributed functionality is realized through distributed networks and object-oriented (OO) design methodologies. This decreases system costs and provides a means with which to deal with the increasing complexity of these systems. Industries such as the automotive, aeronautical, rail, medial equipment and industrial machinery that already use low-level controls would benefit from this advancement by allowing a more flexible and more responsive systems to be designed. In addition to existing industries, the field of Holonic Manufacturing Systems (HMS) would be advanced by allowing for the integration of higher-level agents and low-level agents for a real-time system [13].

A problem arises however, when this model is applied to safety-critical systems because safety-critical systems require guaranteed communication delays. Two opposing paradigms currently exist for dealing with communication: only one of them is currently able to meet this requirement.

The two paradigms commonly used in distributed control systems are either time-triggered or event-triggered [14]. The time-triggered paradigm uses a predefined cyclic time schedule (periodic) to drive the communication of the system. The event-triggered paradigm relies on non-periodic (sporadic) stimuli to generate the need to communicate. To date, the time-triggered approach has been favoured in safety-critical systems because it can ensure that no communication delays occur by assigning dedicated communication time windows to specific nodes [15]. The purely event-triggered approaches have been unable to guarantee communication delays and have thus not gained acceptance.

However, work has been done to make event-triggered systems more predictable [16]. The problem with the approach is that they have converted event-triggered communication into time-triggered communication by placing a minimum time period between successive communications. But this essentially violates the idea that events take place independent of any time reference.

This paper contrasts time-triggered and event-triggered paradigms in section 2. Section 3 introduces a new paradigm and a new methodology to compare protocols. Section 4 reviews the recent findings related to the new paradigm. Section 5 concludes by reviewing the accomplishments and future research potential.

2. Existing Comparisons

Architectures originally conceived for aircraft applications are now being used in the automotive, railway and industrial equipment industries. With the introduction to these new markets, economies of scale has driven down component prices during the 1990's and a wide variety of bus architectures are now available. Competing architectures include time-triggered buses such as SAFEBus, Time-Triggered Controller Area Network (TTCAN), TTTech Time-Triggered Architecture (TTA), and FlexRay and event-triggered buses such as Controller Area Network (CAN), ByteFlight, and LonWorks [17, 18, 19, 20]. In fact most of these architectures attempt to combine some form of event-triggered and time-triggered communication.

2.1. Communication Paradigms

A typical time-triggered protocol defines a network cycle during which each node is given a dedicated time window during which to it may broadcast a message. Each cycle begins with a reference message that contains the schedule for the upcoming cycle. Time-triggered messages are scheduled into exclusive windows, to ensure that there are no collisions. With no collisions occurring, the time-triggered approach is able to guarantee delivery time, and hence be deterministic. The downside is that much of the communication bandwidth is lost when a node does not transmit during a given time window.

An event-triggered protocol communicates messages as a response to stimuli, which occur at any time. As a result collisions occur and how they are handled is a key concern. Some protocols require the retransmission of both messages as with the Ethernet, while others employ various arbitration techniques. In either case, at least one of the messages will not be sent immediately, and is delayed in its transmission. This is a key characteristic of all event-triggered protocols. The CAN protocol uses a bit-wise arbitration technique that ensures that one of the two (or more) messages that collide is transmitted without any further delay. Regardless of the arbitration, all event-triggered protocols experience increasing delay with increasing traffic branding them non-deterministic.

2.2. Communication Schedule

The time-triggered approach developed naturally from the traditional closed-loop control systems. Traditional control systems operate by polling sensors on a cyclic basis and responding as required. Similarly, the time-triggered model relies heavily on a periodic internal time schedule for communication.

The event-triggered approach evolved out of the need to handle sporadic communication events such as alarms. Because nodes respond to external stimuli, which occur at any time, no time schedule or synchronization is required.

2.3. Communication Measure

A system's communication may be classified as either deterministic or probabilistic. Determinism in control systems is described by Dimyati K. (1996) [21]:

"A communication system is said to be deterministic if, when operating within its stated maximum load, all messages with a defined priority can be delivered within a specific time period."

Probabilistic systems, on the other hand, cannot guarantee message delivery in a timely manner, but only the probability that is will occur.

Time-triggered protocols are typically deterministic and as a result, their proponents typically emphasize the need for a system to be predictable. To them guaranteeing message delivery time is more important than response time when measuring a system's performance.

Event-triggered proponents on the other hand, place an emphasis on response time. This is done because event-triggered systems can only describe message delivery time probabilistically because of the uncertainty due to message collisions. The event-triggered protocols however, can have very fast response times and thus the preferred measure of a system tends to focus on the potential response time and bandwidth efficiency.

2.4. Communication Evaluation

Because of the different capability and focus of the supporters of each paradigm, there results a difference in general evaluation techniques.

Event-triggered systems tend to be evaluated based on the overall performance of the system. This evaluation can be effectively measured by a simulation of the operation of a real system under various conditions.

Time-triggered systems are often evaluated based on their ability to be deterministic. The evaluation is done by considering the worst-case performance of the system. Typically this is done by considering the lowest priority node during times of peak traffic. From this analysis it is shown that the system can guarantee that the communication delay never exceeds a certain value, and is hence deterministic.

2.5. Communication Summary

The two opposing paradigms presented above result in two different communication schedules, performance measurement emphasis' and evaluation techniques as summarized in Figure 1.

Paradigm	Event-Triggered	Time-Triggered
Schedule	Sporadic Source	Periodic Source
Performance Measure	Response Time	Delivery Time
Evaluation Technique	Simulation	Worst Case Analysis

Figure 1. Communication Comparison

3. Re-evaluation

The comparison of the previous paradigms reveals a large difference in how each paradigm has developed, how they schedule communication, how their proponents favor measurements, and how different techniques are used. It should be no surprise then that there have been a large number of protocols that have attempted to "bridge the gap" between the two extremes. Examples of these hybrid protocols are Time-Triggered CAN (TTCAN), Flexible Time-Triggered CAN (FTT-CAN), ByteFlight and FlexRay. Each of these protocols attempt to accommodate both event-triggered and time-triggered traffic. This is typically done by reserving a time window in which each type of communication scheduling can occur.

This has been proven to be a very appealing way of getting the best of both worlds and in fact was one of the influences that lead to a completely different approach. That approach, shared next, also yields the best of both worlds, but does so not by combining two paradigms, but by inventing a new one.

3.1. New Paradigm

A new paradigm for scheduling communication has been previously introduced [22]. In the new paradigm, bus access is the same as a Controller Area Network (CAN) with the exception of how priorities are set. In typical CAN systems, priorities are static and do not change during operation. In the proposed **dynamic-priority** protocol, priorities are assigned based on the nodes message broadcast history.

This method involves two components. The first component, the *Time Priority (TP)*, is a calculation based on the message history within the node. The second component, the *Node Priority (NP)*, is a unique static identifier like the one used in the fixed priority scheduling method. Together these two components make up a *Dynamic Priority (DP)* code that is computed by each node.

The *Time Priority* component is used to increase a node's priority the longer it waits and is based on the time since the node last sent a message. An important feature of this is that no clock synchronization is required between the nodes as is the case with time-triggered systems.

The *Node Priority* component is assigned pre run-time as a unique identifier for each node. It is required when two messages collide that have the same time priority. This happens only when two nodes that have not transmitted for greater than the cycle time attempt to transmit at the same time. In this case both nodes would have the highest time priority, and the static component would then be used to settle the arbitration.

Table 1. Example of an Escalating Priority Schedule Over Eight Time Steps

Node #	Escalating Priority Value							
1	21	*11*	--	--	--	--	--	*11*
2	32	22	*12*	--	--	*32*	52	42
3	43	33	23	*13*	--	43	*33*	53
4	*14*	--	--	--	--	--	--	--
5	15	15	15	15	*15*	55	45	35
Time Step	1	2	3	4	5	6	7	8
System Time	C	2C	3C	4C	5C	6C	7C	8C
Broadcasting Node	4	1	2	3	5	2	3	1

3.2. Dynamic Schedule

One of the important keys of the new paradigm is that it can handle both sporadic and periodic communication. Traditionally, periodic message schedules can be determined off-line or pre-run-time because the periodic cycle time is known before hand. Sporadic messages on the other hand, are typically scheduled as they arrive in real-time through some type of arbitration. Because the two types of messages are scheduled differently, a protocol would have to employ two different types of scheduling to handle both types of messages.

The protocol introduced here is able to schedule both types of messages using only one scheduling method. This is achievable because the scheduling method is able to maintain a deterministic delay time even when scheduling the periodic messages in real-time. As a result, both periodic and sporadic messages are able to be scheduled **on-the-fly**.

3.3. Refined Measurement

The current gap between the two communities that favour time-triggered and event-triggered approaches, stems from an unclear definition of requirements for a safety critical communication. It is felt that a refined quantitative definition of how responsiveness is measured will provide a proper benchmark against which all systems can be evaluated. Measuring responsiveness alone leaves the door open to criticism from the safety-critical community and their need for predictability. Measuring only the worst-case delays hides the inherent inefficiency and lack of flexibility.

Instead, the new measurement methodology should be to qualifying a system given the strict deterministic requirements, but then move on to evaluate the system's responsiveness in terms of its **average and maximum cycle time**s.

3.4. Additional Evaluation Technique

The existing evaluation techniques are completely valid and should continue to be used. The analytical worst-case analysis is capable of determining if the protocol is deterministic. A discrete event simulation model is well suited to measuring the overall performance of individual nodes of the system to evaluate how uniformly each node performs with respect to the others. In addition however, another method is proposed to augment the descriptiveness of the existing methods.

A **game theory** method may be used to describe the communication negotiation that takes place between the nodes. The game theory approach is being evaluated because of its ability to measure competition and cooperation between players for a resource. In the case of system evaluation, the nodes on a network represent the players, and they are competing for a resource which is the bus.

3.5. Summary

As summarized in Figure 2, the newly suggested paradigm shifts the focus from developing the two paradigms separately to attempting to incorporate the strengths of both into a single system. The dynamic-priority paradigm shows the potential for unifying two communication schedules into a single protocol. The addition of a qualification step before proceeding to more detailed cycle time analysis will hopefully satisfy proponents on both sides. Finally, the use of game theory is suggested as a potential new method to describe the behavior of a system that was previously unavailable.

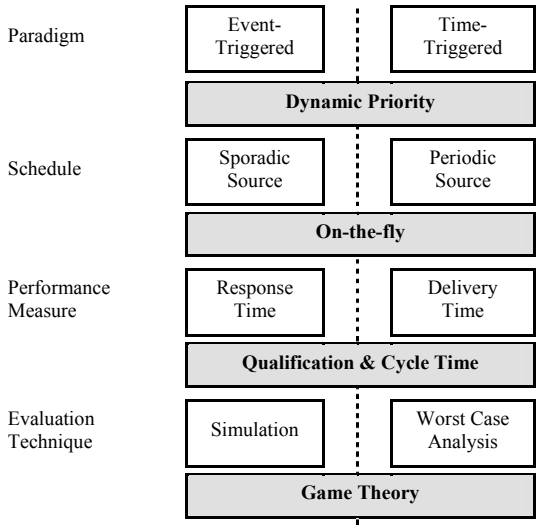

Figure 2. Communication Comparison

4. Initial Results

Initial research has yielded some promising results. A discrete event simulation was used to validate the behavior of event-triggered and time-triggered systems. The behavior of the dynamic-priority system was then modeled for comparison purposes. A worst-case analysis

was then done in order to qualify the dynamic-priority system as being deterministic. Finally, a game theory scenario was developed to test the ability of using game theory to provide an additional method of quantifying the systems. These three initial research directions are described in more detail next.

4.1. Simulation

Initial simulation results for the time-triggered, event-triggered and dynamic-priority scheduling methods are consistent with the expected outcome [22]. In each case the nature of the simulation delay time is consistent with the commonly expected nature of the scheduling method.

The time-triggered protocol is shown to be deterministic. This is evident by the constant maximum delay throughout Figure 3. This delay coincides with the total cycle time for one entire cycle of communication windows.

In contrast, the event-triggered schedule shows that the delays increase unbounded as the network traffic increases as expected. The results also confirm that at lower network loading levels, the event-triggered protocol has a lower delay time than the time-triggered protocol. This is evidence of the faster potential response time.

The dynamic-priority scheduling method achieves two goals. First, it performs deterministically because it never exceeds the delay time of the time-triggered protocol. Secondly, the average delay time is better than those shown by the time-triggered method at low to mid network levels. At high network traffic levels, the protocol's delays appear "clipped". This is a result of the protocol's unique ability to adapt to high network loading situations.

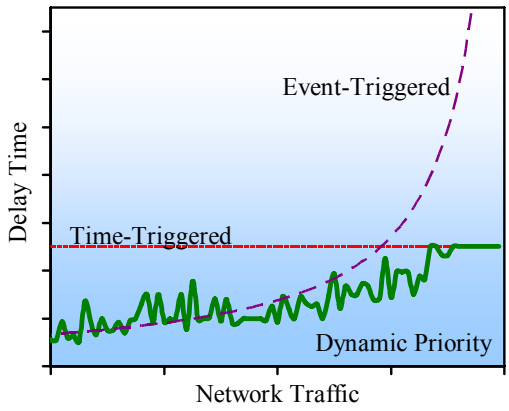

Figure 3. Simulation delay times

4.2. Worst Case Analysis

Dynamic-priority scheduling allows the maximum delay time to be deterministically predicted [23]. In order to achieve this, the priority setting scheme must guarantee that each node will not have to wait longer than one cycle to broadcast. This maximum delay time is equivalent to the delay time a time-triggered protocol can guarantee.

The dynamic-priority scheduling proposed involves two components. The first component is a calculation based on the message history within the node. The second component is a unique static identifier like the one used in the fixed priority scheduling method. Together these two components make up a dynamic-priority code that is computed by each node.

The calculated component is used to increase a node's priority the longer it waits and is based on the time since the node last sent a message. The static component is assigned pre run-time as a unique identifier for each node.

The analysis presented demonstrates that a node is guaranteed to transmit a message once every cycle [23]. This guarantee deems the dynamic-priority protocol as deterministic.

4.3. Game Theory

Game theory is an interdisciplinary approach to the study of human behavior that has its roots in mathematics, economics, and the social sciences [24, 25]. A game is an interaction between players who employ various strategies to achieve various outcomes. This fits very naturally with the idea of various nodes competing to send a message on a network. In this case the players are the nodes and the strategies they employ will allow them to (or not allow them to) send a message.

A brief recap of the initial work is provided here before the benefits are discussed [26].

Payoffs for a node are a measure of the benefit to the node given a particular outcome. If for example, one node is able to send its message and another competing node is not, the first node would have a greater payoff than the latter. The bimatrix payoff format is then used to compare the various communication strategies.

Various paradigms were evaluated, and the results allow for a comparison of fairness between the paradigms as well as an indication of bandwidth efficiency. By further adding a weighted evaluation, details about the bandwidth usage and individual bandwidth usage provided further insight into the fairness of the scenario while preserving information about its efficiency.

From Figure 4 below, it can be seen that the event-triggered and dynamic-priority protocols have superior bandwidth utilization that the time-triggered protocol.

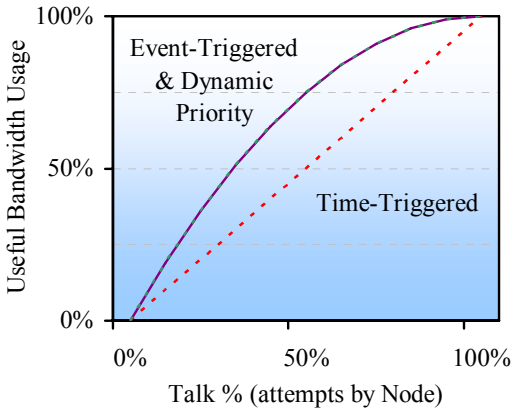

Figure 4. Bandwidth usage comparison

From Figure 5, it can be seen that the event-triggered protocol does not show fairness between the two competing nodes. This result is expected because one of the two node has a static priority level that is higher than the other.

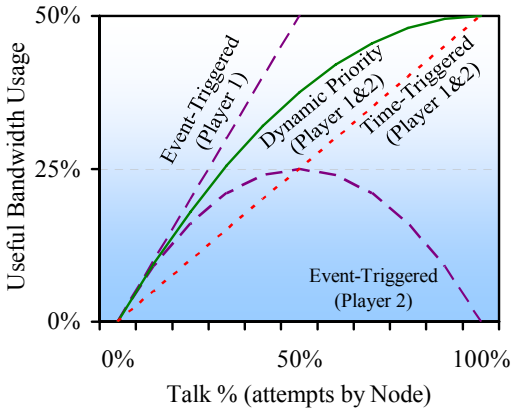

Figure 5. Unequal bandwidth usage

4.4. Summary

The previous sections have demonstrated three conclusions about the dynamic-priority protocol:

- The response time is always better than the time-triggered protocol, and often similar to the event-triggered protocol (as per simulation)
- The dynamic-priority protocol is as deterministic as the time-triggered protocol (as per worst case analysis)
- The bandwidth usage is more efficient than the time-triggered protocol and fairer than the event-triggered protocol (as per the game theory payoffs)

These conclusions can all be made while still providing support for sporadic and periodic messages.

5. Conclusion

The dynamic-priority paradigm is a promising new approach to scheduling low-level communications. Each of the three initial research directions (simulation, worst-case analysis, and game theory) has warranted further research. Future plans are to expand the simulation to accommodate communication errors like babbling idiots and noise. The worst-case analysis will be expanded for systems with unequal messages lengths. This analysis may then be able to be expanded to describe the worst-case response for nodes other than the lowest priority node. The game theory approach leaves the most room for development. Expanding from a 2 player game to an N player game and allowing players to choose different strategies will be explored.

But the most promising future work may come from a further refinement of the dynamic-priority paradigm itself. Just as there are many event-triggered and time-triggered protocols that have continued to be developed and improve over previous generations, the dynamic-priority paradigms will do the same. There is currently research under way into a dynamic-priority protocol that will be able to schedule sporadic and periodic with varying levels determinism.

6. References

[1] H. Parunak, "Autonomous agent architectures: a non-technical introduction," Industrial Technology Institute Report, 1993.

[2] W. Shen and D. Norrie, "Agent-based systems for intelligent manufacturing: a state-of-the-art survey," Knowledge and Information Systems, 1, 129-156, 1999.

[3] P. Valckenaers, H. Van Brussel, L. Bongaerts, and J. Wyns, "Holonic manufacturing systems," Integrated Computer Aided Engineering, 4, 191-201, 1997.

[4] S. Olsen, J. Wang, A. Ramirez-Serrano, R.W. Brennan, "Contingencies-based reconfiguration of distributed factory automation," Robotics and Computer-integrated Manufacturing, 21(4-5), pp. 379-390, 2005.

[5] J. Christensen, "HMS/FB architecture and its implementation," In: M. Deen (Ed.) Agent-based Manufacturing: Advances in the Holonic Approach, pp. 53-88, 2003.

[6] T. Strasser, A. Zoitl, F. Auinger and C. Sunder, "Towards engineering methods for reconfiguration of distributed real-time control systems based on the reference model IEC 61499," In: V. Marik, R.W. Brennan, M. Pechoucek (Eds.), Holonic and Multi-agent Systems for Manufacturing, Lecture Notes in Computer Science, Vol. 3593, pp. 165-175, 2005.

[7] B. Douglass, Doing hard time: Developing real-time systems with UML, objects, frameworks, and patterns, Addison-Wesley, 1999.

[8] R.W. Brennan, M. Fletcher, S. Olsen, D.H. Norrie, "Safety-critical software for holonic control devices," Proceedings of the IMS International Forum on Global Challenges in Manufacturing, Cernobbio, Italy, pp. 767-774, 17-19 May, 2004.

[9] V. Vyatkin and H. Hanisch, "Component design and formal validation of SFA systems: a case study," 5th IEEE/IFIP International Conference on Information Technology for Balanced Automation Systems in Manufacturing and Services, Cancun, Mexico, pp. 313-322, 25-27 September, 2002.

[10] S. Bussman and J. Sieverding, "Holonic control of an engine assembly plant: an industrial evaluation," Proceedings of the 2001 IEEE International Conference on Systems, Man, and Cybernetics, Tucson, AZ, 2001.

[11] P. Vrba, F. Macurek, and V. Marik, "Using radio frequency identification in agent-based manufacturing control systems," In: V. Marik, R.W. Brennan, M. Pechoucek (Eds.), Holonic and Multi-agent Systems for Manufacturing, Lecture Notes in Computer Science, Vol. 3593, pp. 176-187, 2005.

[12] P. McGuire, G. Liggins, A. Benaskeur, R.W. Brennan, and P. Wojcik, "The application of holonic control to tactical sensor management", Proceedings of the IEEE Workshop on Distributed Intelligent Systems, 2006 (in press).

[13] Holonic Manufacturing Systems, *Web Site*, http://hms.ifw.uni-hannover.de/, 2004.

[14] A. Albert, and Robert Bosch GmbH. "Comparison of event-triggered and time-triggered concepts with regard to distributed control systems", *Embedded World*, Nürnberg, 2004.

[15] J. Rushby, "A comparison of bus architectures for safety-critical embedded systems", CSL Technical Report, SRI International. 2001.

[16] K. Tindell, A. Burns, and A.J. Wellings. "Calculating controller area network (CAN) message response times", Control Engineering Practice, 1995; 3(8): 1163-1169.

[17] Robert Bosch GmbH., "Bosch CAN Specification version 2.0", 1991.

[18] P. Ferreira, Pedreiras, Almeida and Fonseca, "The FTT-CAN protocol for flexibility in safety-critical systems," IEEE Micro, 2001, pp. 81-92.

[19] H. Kopetz, "A comparison of TTP/C and FlexRay," TU Wien Research Report, 2001/10.

[20] M. Peller, J. Berwanger, and R. Griessbach, "Byteflight Specification Draft Version 0.5", *Website,* Retrieved from <http://www.byteflight.com>, 1999.

[21] K. Dimyati. "Real-time clock syncronisation over a CAN-based fieldbus", PhD Thesis, Electrical & Electronic Engineering Dept., University of Wales, Swansea, U.K. 1996.

[22] J.J. Scarlett, R.W. Brennan, and D.H. Norrie, "A proposed high-integrity communication protocol for real-time control," *Proceedings of the IMS International Forum on Global Challenges in Manufacturing*, Cernobbio, Italy, pp. 759-766, 17-19 May, 2004.

[23] J.J. Scarlett and R.W. Brennan, "An event-triggered communication protocol for intelligent real-time control," *Proceedings of the 16th IFAC World Conference*, Prague, Czech Republic, 4-8 July, 2005.

[24] G. Owen, "Game Theory", 2nd Edition, *Academic Press*, New York, 1982.

[25] M.J. Osborne and A. Rubinstein, "A Course in Game Theory", *The MIT Press*, Cambridge, Massachusetts, 1996.

J.J. Scarlett and R.W. Brennan, "A New Evaluation Method of Communication for Distributed Control," *Proceedings of the IEEE Workshop on Distributed Intelligent Systems*, Prague, Czech Republic, June 15-16, 2006.