The Development of a Simulation Model for the Flexilink Protocol

Dalia Hassan El-Banna, Yonghao Wang, Michael Clarke and Sharon Cox
Faculty of Technology, Engineering & the Environment,
Birmingham City University
Corresponding e-mail sharon.cox@bcu.ac.uk

Abstract: *The Flexilink protocol has been proposed to support audio-visual communication across the Internet. The protocol aims to support low latency to reduce communication jitter and support different sampling frequencies. This paper discusses real-time Ethernet protocols and presents a simulation model developed to evaluate the Flexilink protocol using the OMNeT++ simulation platform to simulate the behaviour of the Flexilink protocol over Ethernet networks. Simulation-based evaluation strategy enables the protocol to be evaluated by considering the effects of different parameters and different topologies, which would have been both time-consuming and costly if done using real hardware.*

Keywords: OMNeT++, Flexilink, Real-time, Protocol, Simulation, Low Latency, Audio-Visual Communication

Introduction

There is an increasing demand to support interactive audio/video media traffic over the Internet. Interactive audio/video communication is based on sending real-time traffic. This type of traffic requires low latency in addition to preserving the time relationship between packets of the same flow (low jitter) (Austerberry, 2005). However, a connectionless packet switched network architecture with the current techniques used for QoS and traffic engineering, is not suitable for supporting this type of deterministic traffic especially when low latency and low jitter is required (Wang, et al., 2012).

In 2010, the Flexilink protocol was introduced, offering a unified network solution to support both best effort traffic and low latency deterministic traffic (real-time traffic) (Grant, 2010). In addition, Flexilink supports multi audio channels with different sampling frequencies and word lengths; a feature that is not available in current multiplexing schemes without sampling rate conversion and data format rectification (Wang, *et al*.,2012).

This paper outlines the development of a simulation model for the proposed Flexilink protocol using OMNeT++ simulation platform to simulate the behaviour of Flexilink protocol over Ethernet networks. The following section reviews real-time Ethernet protocols to compare the features of the newly proposed Flexilink protocol with features of peer protocols in terms of providing real-time properties to widely used Ethernet networks. The design and implementation of custom Flexilink nodes in the OMNeT++ simulation platform is then discussed in section 3. Section 4 evaluates the simulation model developed. The paper concludes by discussing future work to refine and validate the simulation model.

Real-time Ethernet Protocols

Being a newly introduced protocol, Flexilink does not have an extensive literature-base. However, a number of protocols have been developed to support both time-deterministic as well as best-effort data in order to provide a unified platform for automation networks. Audio/video streaming requires the same time constraints in data transmission as industrial automation networks that involve real-time data transmission for safety critical applications. Thus some real-time Ethernet protocols have been used both in automation and audio/video streaming over Ethernet. As an example, IEEE 802.1 AVB has been validated in previous work to be used both in automation in future in-car networks (Steinbach et al., 2012), as well as in data steaming over Ethernet (Lim, *et al*., 2011). This section reviews some of the existing real time Ethernet protocols.

Real-Time Ethernet Background

Real-time electronic control systems have been employed to control and monitor safety-critical applications in the avionics and automotive industries. The demand for having physically distributed control in strict real-time requires network protocols that support stringent real-time requirements as well as a guarantee of service to ensure that they will always operate deterministically and correctly (Doyle, 2004). Ethernet was not suitable to provide this kind of networking, being non deterministic. As defined in IEEE 802.3, Ethernet uses CSMA/CD as a media access control protocol which results in possible failure of transmission and random delays due to the backoff algorithm. However, Ethernet technology has proven to be the most successful and dominant Local Area Network due to its flexibility and the fact that it is a highly scalable protocol. Also Ethernet is able to support the TCP/IP stack, which offers an attractive feature for existing automation networks in which devices can easily gate to the Internet. This feature allows remote monitoring and control of the network. In addition, Ethernet offers increased bandwidth compared to other real-time solutions such as PROFIBUS which offered 12Mbps compared to 100Mbps or probably 1Gbps offered by using Ethernet. These benefits increased the demand for developing protocols that provide real-time characteristics to the existing Ethernet standard. These newly developed set of protocols are referred to as Industrial Ethernet or Real-Time Ethernet. In the next section a critical literature review is conducted for some of the most used real-time Ethernet protocols. A full list of real-time Ethernet protocols can be found in Schwager (2003).

2.2 Real-Time Ethernet Protocols

Popp and Wenzel (2001) introduced PROFINET based on the IEEE 802.3 Ethernet standard and is interoperable with TCP/IP. It was first used for distributed automation systems and was compatible with the existing standard used for interconnecting devices. In PROFINET, different types of data are sent over the same channel using TDMA with a highly precise synchronized cycle based on the IEEE 1588 standard for precision clock synchronization protocol (Popp and Wenzel, 2001).

Ferrari *et al.* (2004) evaluated the performance of real-time and non-real-time classes of PROFINET traffic. Results showed that using PROFINET non-real-time protocol in an unloaded network resulted in a maximum delay of less than 500µs with standard deviation (jitter) of 50µs. However, adding an FTP server to the network increased delay with a maximum delay of 156ms with an average of 21ms and standard deviation of 27ms.

With a cycle time of 32ms, the average delay between successive transmission was 32.107ms with standard deviation of 1ms in the case of a loaded network. Ferrari et al., (2004) argued that this calculated standard deviation is less that 5% of the cycle time (32ms) and thus concluded the correct performance of the PRROFINET-RT protocol for supporting hard real-time control systems.

Being a hardware based solution, PROFINET-IRT achieved delays in the range of 1ms with standard deviation of 25µs. Thus PROFINET-IRT was suitable to use with systems requiring hard real-time performance, typically, high performance motion control systems (Doyle, 2004). A proposal for enhancing the performance of PROFINET-IRT in terms of delay was introduced by Schumacher et al. (2008), where an attempt was made to decrease the delay in the range of below milliseconds. Schumacher et al. (2008) suggest that the main components influencing the performance regarding the minimum achievable delay are the propagation time and frame transmission time.

Since the propagation delay is mainly affected by the transmission medium which cannot be significantly changed, Schumacher *et al*. (2008) proposed different mechanisms to optimize the frame transfer time. Firstly, a topology-based forwarding algorithm using local MAC addresses was introduced to reduce the time spent for address look-up and thus the time taken in the forwarding process. Secondly, a mechanism was introduced to decrease the length of the used preamble bytes to two bytes (1 preamble + 1 SFD) instead of eight bytes used in the standard Ethernet frame. This was found to improve the overall frame transfer time by 480ns (Schumacher *et al*. 2008).

Another real-time Ethernet protocol, EtherCAT, was developed by Beckhoff in 2003 (Prytz, 2008). EtherCAT operates in a master/slave environment with the master initiating data transmission by sending an Ethernet frame to the slave nodes. EtherCAT uses a summation frame approach where a single Ethernet frame is used to deliver data (referred to as EtherCAT telegrams) to more than one node (slaves). Each node extracts the data from the frame addressed to it, puts some new data in the frame and then sends the frame to the next slave. All the message reception, data processing and frame retransmission operations are made "on the fly" by the nodes, without any extra delay (Seno and Zunino, 2008). Using summation frames increased EtherCAT bandwidth efficiency and being a hardware based protocol helped achieving delays in order of microseconds. Seno and Zunino (2008) developed a simulation model for EtherCAT using OPNET. Simulation results from the developed model were compared with the theortical analysis and confirmed that delay increases as the number of slave nodes on the network increases.

Prytz (2008) compared the real-time performance of EtherCAT and PROFINET-IRT using scenarios on both 100Mbps and 1Gbps Ethernet. Results showed that EtherCAT had better real-time performance compared to PROFINET-IRT. Prytz (2008) suggest that these results conform to the expected performance as the EtherCAT communication mechanism relies on the summation frame principle where a single frame is used to accommodate data for a great number of slave nodes. In contrast, PROFINET-IRT needs to send a packet per each addressed network node. Due to this feature, the EtherCAT protocol was considered to have the highest communication efficiency among real-time Ethernet protocols. It was considered particularly suitable for networks with a large number of devices with small payloads (Knezic *et al*. 2011).

EtherCAT can be problematic in large scale networks that consist of several hundreds of devices (Knezic et al., 2011) because the EtherCAT protocol uses the logical ring topology where network propagation time is greatly affected by the number of network nodes. Knezic *et al.* (2011) proposed an approach to improve EtherCAT efficiency over a large scale by exploiting the symmetric spatial distribution property of the EtherCAT networks in which input/output data can be exchanged more efficiently leading to a significant decrease in EtherCAT frame transmission time and thus increased communication efficiency. However, a drawback of the proposed approach is that it is not suitable for networks where topology changes are not allowed or in the case of non-symmetrical networks.

Time-Triggered Ethernet (TTEthernet)

Another real-time Ethernet protocol is the Time-Triggered Ethernet Protocol (TTEthernet) (Steiner, 2009) that offers deterministic real-time communication and TCP/IP Ethernet traffic in parallel on the same network. TTEthernet's high fault-tolerance and high availability mean that it is appropriate for use in safety critical aplications such as in aerospace system design and automotive systems (Steiner, 2009).

The main concept of TTEthernet is that it uses periodic cycles to transmit time-critical traffic (referred to as Time-Triggered (TT) traffic). Each node in the network is assigned an offline predetermined timeslot to send its real-time traffic. This approach ensures predictable transmission delays with no queuing and thus low latency and jitter (Steinbach et al., 2011). In addition to the time-triggered traffic, TTEthernet defines another two types of traffic, Rate-Constraint Traffic (RC) and Best Effort Traffic (BE). Rate-constraint traffic is time critical traffic sent with less rigid temporal requirements than time-triggered traffic. Rate-constraint traffic is based on the AFDX standard (Condor Engineering, 2005). Best effort traffic is the standard Ethernet traffic and is sent with the lowest priority. To differentiate between these classes, TTEthernet uses a content-oriented addressing format where the 48-bit destination Ethernet MAC address is used. The MAC address is divided into two parts. The first part is the Critical Traffic Identifier (CT-ID) where each message has unique CT-ID used for routing. The second part of the MAC address is the Critical Traffic Mask (CtMask). Both CT-ID and CtMask formsthe CtMarker that is used to differentiate critical time traffic (TT and RC) from the best effort traffic.

A simulation model for TTEthernet full operation was introduced by Steinbach et al., (2011). The proposed model, TT4INET (CoRE, 2012) presented an extension for the OMNeT++ INET framework. The implemented model provided modules for TTEthernet host as well as TTEthernet switch.

IEEE 802.1 AVB

Real-time Ethernet protocols are also used in real-time data steaming (audio/video) to provide a unified network for transmitting time determistic as well as best-effort traffic. IEEE 802.1 AVB protocol introduced different mechanisms to enable time-synchronized low latency streaming services through 802 networks specified by the AVB task group (IEEE802.org, 2011). Standards used by AVB are:

- IEEE 802.1AS: Time synchronization protocol to enable synchronization of distributed nodes in switched Ethernet . Synchronization is achieved with an accuracy of less than 1 μs over maximum seven hops.

- IEEE 802.1Qav: Specifies queuing and forwarding of time critical traffic. AVB defines two classes of traffic, in addition to the best effort, depending on the required end-to-end delay:

- Stream reservation (SR) class-A: Guarantees a maximum 2ms end-to-end delay.

- SR class-B: Can achieve up to 50ms end-to-end delay.

- IEEE 802.1Qat: Signaling mechanism for resource reservation to ensure that AVB time critical traffic will have the required resources along the entire path from source to destination. Only 75% of the avaliable bandwidth can be reserved whereas the rest is used to forward best-effort traffic.

Lim *et al.* (2011) developed a complete simulation model of the AVB protocol in OMNeT++. The developed model was used to evaluate the performance of AVB protocol in the worst case scenario of seven hops switched Ethernet to verify the time constraints of streaming data modelled as class A and class B traffic.

Steinbach *et al.* (2012) used the TT4INET framework and developed a simulation model for AVB to provide simulation-based performance evaluation of both IEEE 802.1 AVB and TTEthernet when used in an in-vehicle network. Simulation results showed comparable performance for both protocols, however increasing the frame payload of background traffic affected the end-to-end delay when using the AVB protocol. This was not the case however with TTEthernet which showed robust performance even in the presence of large frames background traffic.

Flexilink Operation

Flexilink is a protocol that combines the advantages of time division multiplexing (TDM) and best effort networks (Grant, 2010). It introduces a unified network structure for supporting both time-deterministic traffic, such as audio/video and other data types that require constant transmission intervals and predictable delay, as well as best-effort data. In addition, Flexilink provides full compatibility with existing network architectures and protocols.
According to Wang *et al.* (2012), a node that supports Flexilink protocol classifies traffic into three main categories which are:

- Synchronous Flow (SF) representing time-deterministic traffic such as audio/video.
- Asynchronous Flow (AF) representing best-effort traffic that are sent without real-time constraints but are rather sent at the earliest possible opportunity.
- Control Messages (CM) used for setting up and tearing down the link between two Flexilink nodes.

In order to ensure that SF traffic is complying with timing requirements, time slots are reserved for transmitting SF data packets. This is done by using control messages which can be implemented using IEC 62379-5-2 standard. The standard uses SNMP for monitoring and controlling of networked audio and video (IEC Project Team 62379, 2012). Once the link is established, intelligent time slot maps identify the reserved time slots based on the requirements of the time-critical traffic (synchronous flow), gaps between reserved time slots can then filled with asynchronous flow traffic. This operation of Flexilink can achieve maximum utilization of link capacity where all the gaps, between synchronous flow data packets, are filled with asynchronous flow traffic. Implementing Flexilink over

Ethernet further maximizes the available link capacity and bandwidth efficiency by using Ethernet Jumbo Frames. Jumbo frames have a payload size greater than the standard 1500 bytes Ethernet Frames. Both synchronous flow and asynchronous flow sent data packets are allocated in the payload. This increased payload also helps in reducing the cost headers and inter-frame gap.

Once the link is set up between the two Flexilink nodes, all subsequent frames are sent from the Flexilink node at one end of the link to the node at the other end. This means that both the source and destination fields of the Ethernet Jumbo Frame header are no longer needed. Eliminating these unnecessary fields will further maximize the frame payload to carry more data across the transmission link. This new frame format is referred to as Reduced Jumbo Frame (RJFrame). In RJFrame, each synchronous flow packet will have a 1-byte header indicating the length of the corresponding packet. This design allows Flexilink to support multiple audio channels with different sampling frequencies and different bit rates.

Design of Simulation Model

A model was developed to simulate the behaviour of the Flexilink protocol over Ethernet networks using OMNeT++ simulation platform, INET framework and the real time extension TT4INET.

Simulation Environment

OMNeT++ is an open source simulation tool freely available for research and academic use. It is an object-oriented modular discrete event-based simulation platform. This discrete event-based simulation is suitable for modelling data networks where network behaviour can be simulated by modelling the events in the network such as sending and receiving of a packet.

The INET framework is an extension on the OMNeT++ simulation platform. It has several modules ranging from the physical layer to the application layer of the OSI model. Protocols such as IPv4, IPv6, TCP and UDP are implemented in the INET framework. Link layer protocols such as Ethernet and 802.11 are also implemented.

Simulation models of both TTEthernet and IEEE 802.1 AVB are based on the INET framework for OMNet++. These models have been introduced and validated in previous work such as Lim *et al*. (2012) and Steinbach *et al.* (2011). The source code of TTEthernet, TT4INET, is published by the CoRE (Communication over Realtime Ethernet) Research Group (2012).

Simulation of Flexilink protocol using OMNeT++ and INET framework requires the design of custom nodes. TT4INET components were customized to model the Flexilink protocol.

Flexilink Implementation in OMNeT++

Based on the specification of Flexilink protocol, simulation can be divided into two main stages. During the first stage, the link between the hosts is setup with an intelligent time slot map algorithm. In the second stage, the actual data exchange between hosts takes place again based on Flexilink protocol specification. Custom nodes had to be implemented in C++ to be able to simulate the behaviour of Flexilink protocol using OMNeT++.

The simulation model for Flexilink Protocol is based on both the INET framework (INET, 2012) and the TTETHERNET4INET-Framework. In order to be able to simulate the behaviour of a network node running Flexilink

protocol it was assumed that traffic from N audio sources S1,….,SN with sampling frequencies F1,….,FN and thus sampling periods T1,…,TN respectively. Figure 1 shows scheduling of audio sources traffic with t0 denoting the start of the allocation period and the arrows representing the beginning of the time slots allocated for each audio source. In the simulation model, the cycle time (Tc) is defined to be equal to the duration of the Allocation Period (AP) which is equal to the duration of two successive RJFrames which is 124.96 µs on 1 Gigabit Ethernet as per Flexilink specifications. In the simulation model, N is set to be equal 1, that is only one audio source is sending traffic on the link

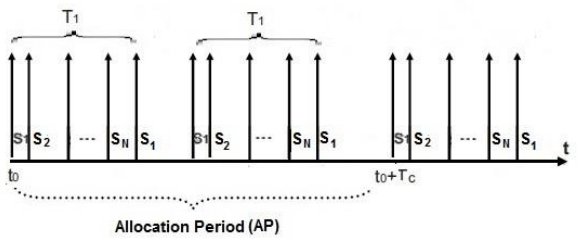


Figure 1: Flexilink Allocation Period (AP)

Network Setup

A simple point-to-point network model with two network nodes was established. The nodes are connected with an error free 1 Gigabit Channel as shown in Figure 2. Each network node (referred to as FlexiHost) is simulated to send traffic generated from two traffic sources. SFTrafficSource models a mono audio source with a sampling frequency 48 KHz and 24 bits encoding generating 1.152 Mbps. To accommodate this bit rate, the source will be allocated six time slots from each allocation period with each slot being five bytes long. The second traffic source, AFTrafficSource, models the background traffic that is sent as best-effort data. Background traffic is modelled as packets of size 1500 bytes which represent the size of standard Ethernet Frame. The time interval between sending packets is uniformly distributed between zero and one second.

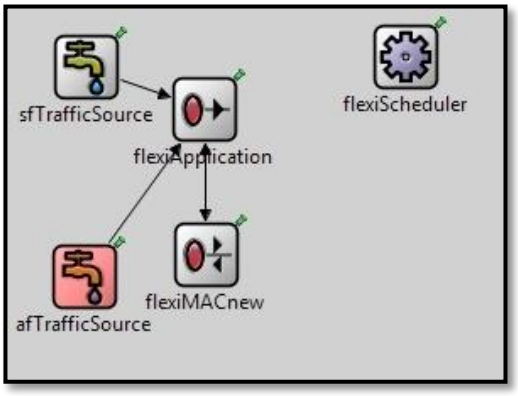


Figure 2: FlexiHost Modules

In the current implementation of FlexiHost however, since only one audio source is considered, only the second type of events is used to trigger the generation of synchronous flow packets to be sent from SFTrafficSource. The first type of events is left for future development of the model where more than one audio source can be considered.

The FlexiHost compound module consists of five modules: FlexiScheduler, SFTrafficSource, AFTrafficSource, FlexiApplication and FlexiMAC. Names gates are self-explanatory. FlexiHost sends RJFrames over the physical link via its gate (referred to as flexigate).

FlexiScheduler Module

The FlexiScheduler module allows events to be scheduled in FlexiHost. FlexiScheduler implementation was based on the design of the scheduler used in TT4INET model. FlexiScheduler measures simulation time in clock ticks. The scheduler module defines two parameters:

- tick: length of clock tick set to 80ns
- cycle_ticks: number of clock ticks in one cycle which is set to 1562 to be equal to the duration of the allocation period 124.96µs.

FlexiScheduler uses the "registerEvent ()" method. Two events are allowed in the FlexiScheduler:

- SchedulerActionTimeEvent: Event that could be triggered at a specific time in the cycle.
- FlexiSchedulerTimerEvent: Event that is triggered after a specific time.

SFTrafficSource Module

SFTrafficSource is a traffic generator for synchronous flow packets in the Flexilink model. To simulate the behaviour of audio traffic in the model, the implemented SFTrafficSource generates equal sized packets at fixed intervals. When "FlexiSchedulerTimerEvent" message is received by the SFTrafficSource on the "schedulerIn" gate, packets are sent to the "FlexiApplication" module through SFout gate. To simulate an audio source with 48 KHz sampling frequency and 24 bits encoding, SFTrafficSource parameters are set as payload equals five bytes and interval equals 20.83µs.

AFTrafficSource Module

AFTrafficSource is a module for best-effort traffic generation. It has the same design as the Ethernet model traffic generator (EthTrafGen module) in the INET framework. The generated packet size is set to be 1500 bytes to simulate full size Ethernet frames. The packets sending interval is set to have uniform distribution between zero and one second. Generated packets are sent to the FlexiApplication module through AFout gate.

FlexiApplication Module

FlexiApplication is a module for outgoing traffic for the FlexiMAC module. The FlexiApplication uses the "handlemessage ()" method for incoming traffic to the module. Data packets from SFTrafficSource are received through SFin gate. Data packets from the AFTrafficSource are received through AFin gate. On arrival asynchronous flow packets are placed in a queue and transmitted in the gaps between the pre-allocated time slots for synchronous flow traffic. When all the payload space (7785 bytes) has been allocated, a frame (FlexiFrame) is then sent to FlexiMAC module through the "out" gate to be encapsulated into the payload of RJFrame and sent on the physical link.

FlexiMAC Module

The FlexiMAC module represents the MAC layer of the FlexiHost node. The FlexiMAC module receives frames (FlexiFrames) from the upper layer (FlexiApplication) via its "upperlayerin" gate. Received frames are then encapsulated into the payload of RJFrame and sent over the physical link. Frames received from the network through FlexiMAC physical gate "phys" are first de-capsulated, removing RJFrame headers, before being sent to the higher layers.

Evaluation of Simulation Model

A simulation of audio traffic transmission was carried out on the designed network. The aim was to compare the results obtained with those expected based on the theoretical results of protocol behaviour.

A simulation test was conducted to validate the performance of the simulation model to make sure that it conforms to the expected Flexilink protocol behaviour. To achieve this, the end-to-end latency of SF traffic sent from Host_A to Host_B on the implemented two node network was measured. Simulation results should show end-to-end delay of synchronous flow

traffic in the order of 3µs to 6µs (per hop) plus the physical medium propagation delay in order to conform to the expected protocol behaviour (Wang et al., 2012). However, during the simulation phase NED files were not linked to their defining C++ classes causing errors in the simulation. Using the latest release of OMNeT++v4.3 (April 2013) should have fixed this bug, however, due to the time constraints of the project, using this version was not possible and validation of the simulation model remains as future work.

5 Conclusion and Future Work

The aim of the project was to develop a simulation model for the Flexilink protocol to serve as a test tool for the future development and study of the protocol. The Flexilink protocol has been carefully examined and a simulation model has been implemented on OMNeT++ simulation platform.

The major limitation in the simulation model is that it is a simplified version of the Flexilink protocol due to time constraints and the challenges encountered in implementing Flexilink in OMNeT++. Further work is needed to:

1. Validate the presented two node network using OMNeT++ v4.3 in terms of conformance of the simulation results with the expected Flexilink behavior presented in Wang et al., (2012).
2. Investigate Flexilink behaviour in transmitting traffic from multiple audio sources with different sampling frequencies and timing requirements.
3. Implement the full version of Flexilink in OMNeT++ including the Precision Timing Protocol (IEEE 1588) and intelligent time slot mapping algorithm.
4. Implement the network switch node that supports Flexilink in OMNeT++ for future implementation of larger networks supporting Flexilink over Ethernet.
5. Compare the performance of Flexilink over Ethernet in steaming data applications to other peer protocols such as IEEE 802.1 AVB.
6. Investigate the potential for using the Flexilink protocol over other physical mediums such as optical fibres.

The model to simulate Flexilink behaviour over Ethernet will help the future development of the protocol. It will also serve as a tool for comparing Flexilink with other peer protocols in terms of performance and design complexity.

References

Austerberry, D. (2005) *The Technology of Video and Audio Streaming,* Second Edition, Focal Press, Oxford.

Condor Engineering (2005) AFDX / ARINC 664 Tutorial (1500-049), http://www.cems.uwe.ac.uk/~ngunton/afdx_detailed.pdf, [accessed 29 September 2013].

CoRE (Communication over Realtime Ethernet) (2012) http://tte4inet.realmv6.org, [accessed 29 November 2012].

Doyle, P. (2004) 'Introduction to Real-Time Ethernet II', *Contemporary Control Systems,* 5(4), pp. 1-6.

Ferrari, P., Flammini, A., Marioli, D. and Taroni, A. (2004) 'Experimental Evaluation of PROFINET Performance', *Proceedings of IEEE International Workshop on Factory Communications Systems,* 22-24 September, Vienna, pp. 331-334.

Grant, J. S. (2010) 'Method And Apparatus For Transceiving Data',

available at: http://patentscope.wipo.int/search/en/WO2010082042, [accessed 23 November 2012].

IEC Project Team 62379 (2012) IEC 62379 Common Control Interface For Networked Digital Audio and Video Products, http://www.cenelec.eu, [accessed 29 September 2013].

IEEE802.org (2011) Audio/Video Bridging Task Group, http://www.ieee802.org/1/pages/avbridges.html, [accessed 22 November 2012].

INET, (2012), 'INET Frameowork', http://inet.omnetpp.org, [accessed 29 November 2012].

Knezic, M., Dokic, B. and Ivanovic, Z. (2011) 'Increasing EtherCAT Performance Using Frame Size Optimization Algorithm', *Proceedings of the IEEE International Conference on Emerging Technolgoies and Factory Automation*, 5-9 September, Toulouse, pp. 1-4.

Lim, H-T., Herrscher, D., Waltl, J. M. and Chaari, F. (2012) 'Performance Analysis of the IEEE 802.1 Ethernet Audio/Video Bridging Standard', *Proceedings of the Fifth International ICST Conference on Simulation Tools and Techniques*, 19-23 March 2012, Sirmione-Desenzano, pp. 27-36.

Popp, M. and Wenzel, P. (2001) 'PROFInet-linking Worlds', *Proceedings of the Eighth IEEE International Conference on Emerging Technologies and Factory Automation,* 15-18 October, Antibes-Juan les Pins, pp. 519-522.

Prytz, G. (2008) 'A Performance Analysis of EtherCAT and PROFINET IRT', *Proceedings of the IEEE International Conference on Emerging Technolgoies and Factory Automation,* 15-18 September, Hamburg, pp. 408-415.

Schwager, J. (2003) 'Information about Real-Time Ethernet in Industry Automation', http://www.pdv.reutlingen-university.de/rte/#Zu Lösung 19: DART-E, [accessed 22 November 2012].

Schumacher, M., Jasperneite, J. and Weber, K. (2008) 'A New Approach for Increasing the Performance of the Industrial Ethernet System PROFINET', *Proceedings of the IEEE International Workshop on Factory Communication Systems*, 21-23 May, Dresden, pp. 159-167.

Seno, L. and Zunino, C. (2008) 'A Simulation Approach to a Real-Time Ethernet Protocol: EtherCAT', *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation*, 15-18 September, Hamburg, pp. 440-443.

Steinbach, T., Kenfack, H. D., Korf, F. and Schmidt, T. C. (2011) 'An Extension of the OMNET++ INET Framework for Simulating Real-time Ethernet with High Accuracy', *Proceedings of the Fourth International ICST Conference on Simulation Tools and Techniques*, 22-24 March 2011, Barcelona, pp. 375-382.

Steinbach, T., Kenfack, H. D., Korf, F. and Schmidt, T. C. (2011) 'An Extension of the OMNET++ INET Framework for Simulating Real-time Ethernet with High Accuracy', *Proceedings of the Fourth International ICST Conference on Simulation Tools and Techniques*, 17-19 March, Lisbon, pp. 375-382.

Steinbach, T., Lim, H-T., Korf, F., Schmidt, T. C., Herrscher, D. and Wolisz, A. (2012) 'Tomorrow's In-Car Interconnect? A Competitive Evaluation of IEEE 802.1 AVB and Time-Triggered Ethernet (AS6802)', *Proceedings of the Vehicular Technology Conference*, 3-6 September, Quebec, pp. 1-5.

Steiner, W. (2009) TTEthernet Specification, http://www.tttech.com, [accessed 10 December 2012].

Wang, Y., Grant, J. and Foss, J. (2012) 'Flexilink: A Unified Low Latency Network Architecture for Multichannel Live Audio', Presented at the 133rd Audio Engineering Society Convention, 26-29 October, San Francisco.