Network Latency and Packet Delay Variation in Cyber-physical Systems

Janette Cardoso, *Member,IEEE,* Patricia Derler, *Member,IEEE,*
John C. Eidson, *Life Fellow,IEEE,* and Edward A. Lee, *Fellow,IEEE*

Abstract

The problem addressed in this paper is the limitation imposed by network elements, especially Ethernet elements, on the real-time performance of time-critical systems. Most current network elements are concerned only with data integrity, connection, and throughput with no mechanism for enforcing temporal semantics. Existing safety-critical applications and other applications in industry require varying degrees of control over system-wide temporal semantics. In addition, there are emerging commercial applications that require or will benefit from tighter enforcement of temporal semantics in network elements than is currently possible. This paper examines these applications and requirements and suggests possible approaches to imposing temporal semantics on networks. Model-based design and simulation is used to evaluate the effects of network limitations on time-critical systems.

Index Terms

Cyber-physical systems, Ethernet networks, Event-triggered, Time-triggered, Synchronization

1. Introduction

The goal of this paper is to illustrate the importance of network temporal semantics in determining the performance of cyber-physical systems (CPS) and other emerging time-critical applications and to discuss candidate techniques for these in existing and yet to be designed network components.

A CPS is a collection of sensors, actuators, computing platforms, and networks deployed to monitor and/or control the properties of an artifact, *the plant*, in the physical world. The passage of time is a critical feature of a CPS. Unlike traditional computing that produces only a succession of system states, a CPS must also measure and in most cases control the time intervals between these states. For distributed CPS, time synchronization is required to form a coordinated view of the state of the physical world and to effect coordinated control over that state.

Many CPS applications include multiple computing platforms, which communicate via networks to control plants with large physical extent. Even when the plant is not physically distributed, networked solutions may be used to distribute computational load, provide physical partitioning of the application, enable more timely local control, or to provide redundancy. The inclusion of networks into a CPS requires that the temporal characteristics of the network be included in the design of the CPS, since network latency and packet delay variation will negatively affect the timing of communications between platforms.

The remainder of this paper is structured as follows. Section 2 provides the background and motivation for the discussion on the role of timing and in particular network timing in the design and implementation of a CPS. Section 3 explores the temporal properties of networks and network elements and possible techniques for evaluating these properties. Section 4 discusses design challenges for CPS posed by networks and considers possible techniques for ameliorating network limitations. Section 5 outlines further work.

2. Background

The ability to accurately assign a timestamp to an event, where the timestamp indicates the physical-time of the event occurrence, is critical in many CPS

- *The authors are with the EECS Dep. of the UC Berkeley. They may be reached at (cardoso, derler, eidson, eal)@eecs.berkeley.edu. Cardoso is also with ISAE, UT France and supported in part by the* French Délégation générale pour l'armement *grant.*
- *This work was supported in part by the Center for Hybrid and Embedded Software Systems (CHESS) at UC Berkeley, which receives support from the National Science Foundation (NSF awards #0720882 (CSR-EHS: PRET), #0931843 (ActionWebs), and #1035672 (CSR-CPS Ptides)), the U. S. Army Research Office (ARO #W911NF-07-2-0019), the U. S. Air Force Office of Scientific Research (MURI #FA9550-06-0312), the Multiscale Systems Center (MuSyC), one of six research centers funded under the Focus Center Research Program, a Semiconductor research corporation program, and the following companies: Bosch, National Instruments, Thales, and Toyota.*

applications. Similarly, the ability to control the time-evolution of a CPS depends on accurately determining the rate of events based on physical-time.

The most commonly used protocols for the distribution of local or standardized time are the Network Time Protocol (NTP) in a LAN environment, global positioning system (GPS) for wide-area environments and specialized, often proprietary protocols in safety-critical systems. For many emerging applications the 1 ms accuracy of NTP is inadequate. GPS is capable of sub-microsecond accuracy but is not suitable for many applications.

Many applications are sensitive to end-to-end transmission delay, *latency*, or to variations in this latency, *path delay variation* (PDV). This section illustrates this point with several important commercial examples and discusses the techniques being specified and deployed to deal with latency and PDV.

2.1. Accurate timestamping in CPS

Accurate timestamps are required in safety-critical systems. These timestamps are typically used to establish time division multiplex (TDMA) communication protocols, cause controlled sampling in devices, and to annotate data for analysis or control purposes. Domain-specific examples are CAN, IEC 61158, ARINC, and TTP, which are widely used in industrial automation, automotive, aircraft, and especially in safety-critical systems.

In the late 1990s, the perceived lower cost, much greater bandwidths, and the non-proprietary aspects of Ethernet resulted in all major vendors of industrial automation and other domains shifting to Ethernet-based communications. However, Ethernet is non-deterministic and can introduce significant latency and PDV which if uncorrected make it unsuitable for many of these applications. This section discusses some of the efforts currently underway to allow sub-microsecond timing to be enforced over Ethernet links in the presence of latency and PDV. Most of these are centered around the IEEE 1588-2008 protocol [1].

Commercial and scientific examples requiring accurate timestamps include:

- Financial: Brokers and other agents linked via Ethernet-based communications require trades to be timestamped with accuracies varying from the millisecond level to the nanosecond level [2].
- Audio-visual: Ethernet-based streaming video and audio in concert halls, homes, business and commercial settings requires timing control for visual and audio quality. Latency is typically not an issue except in applications such as telesurgery and robotics where response time or visual or audio feedback is important. Differences in latency and PDV can be overcome by suitable buffering and reassembly in precise time order based on accurate timestamps applied at the source.
- Trilateration applications: Location of a target such as a gunshot can be derived from several sensors with known locations timestamping the reception of a target signal. For sound, accuracies of a millisecond are required [3] while locating RF transmitters requires accuracies of a few nanoseconds.
- Scientific: The large hadron collider requires timing accuracies at or below the nanosecond level [4].
- Power industry: The industry is moving to increase coverage and accuracy of grid timing via technologies such as synchrophasors [5]. Ethernet is the preferred communications protocol for both substation and long haul communications and industry standards call for timestamp accuracies of $\pm 1 \mu$s.

To meet the needs of these applications, industry is specifying and deploying a variety of technologies to enable accurate timestamping in Ethernet-based systems.

In a networked device, the ability to accurately timestamp network traffic is degraded by fluctuations in the timing of the device's protocol stack and operating system which limits the effectiveness of clock synchronization protocols. The solution is to generate message timestamps for timing protocols, at the bottom of the Ethernet protocol stack. Commercial silicon is available that implements a physical clock, timestamping capability, and some measure of application support either at the MAC or the PHY level. These PHY chips, which are between the MAC and the network media, typically support timing resolution to 8ns or better. Although these chips have been designed with the IEEE 1588 protocol in mind, they are (so far) sufficiently general that they can be used with any Ethernet-based time transfer protocol ([6], [7]).

All LAN-based clock synchronization protocols attempt to measure the path latency between devices which if uncorrected will result in an offset between clocks. PDV degrades the precision of the results. As will be seen in Section 3, PDV in a LAN environment results from queuing in the network bridges and changes in the path topology. Path asymmetry, i.e. the difference between forward and reverse latency, also introduces clock offset. To ameliorate the effects of latency and PDV on synchronization protocols the industry specifies two types of devices: IEEE 1588

boundary and transparent clocks.

A boundary clock terminates and reissues timing traffic, thereby eliminating bridge queues. A transparent clock measures the time a packet takes to traverse the bridge and provides this information to downstream devices to correct for bridge queue delays. These devices eliminate the effects of bridge queues on timing packets, thereby enabling high accuracy and *network traffic independent* clock synchronization. They are not useful in eliminating PDV for ordinary traffic, but the presence of accurate synchronized clocks at all devices allows the measurement of actual end-to-end delays on a per packet basis. This information can be used in some applications to overcome the effects of PDV and latency.

There are also other, specialized synchronization protocols under development and standardization, which attempt to remove or reduce the effects of PDV. For example, the time-triggered Ethernet scheme introduced by Kopetz [8] is being standardized by the Society of Automotive Engineers (SAE).

2.2. Accurate rate control in CPS

The importance of rate control is well illustrated by telecommunication operators providing synchronous T1 (1.544 MBit/s) service over asynchronous Ethernet links. Problems arise for example if the source is transmitting at a higher rate than the sink can accept causing information to be lost. One solution is to provide buffers to accommodate rate discrepancies. To keep buffer sizes manageable, it is necessary to impose strict rate requirements on the telecommunications system even in the presence of Ethernet links. For example, the International Telecommunication Union (ITU) specification on primary clock rate stability in such systems is one part in 10^{11} or one frame slip in 70 days on a 2 MBit/s links [9]. Rate and time control is also critical in cellular backhaul and other telecommunications services [10] and in efforts to incorporate Ethernet into metropolitan area networks.

In matching the frequencies of two distributed clocks, latency itself is not an issue however PDV directly degrades the frequency transfer. The telecommunication industry has major programs devoted to metrics for PDV [11] as well as algorithms for correcting for the effects of PDV [12]. These are very difficult tasks, since the PDV is known to vary with network traffic patterns. Much of this effort is under the auspices of the ITU-T SG15 Q13/15 committee, but there is also a great deal of industrial work and some products incorporating PDV filtering algorithms for use with IEEE 1588 [13].

The ITU also standardized a physical layer solution called *synchronous Ethernet* or *SyncE* for frequency transfer. Synchronous Ethernet devices recover frequency from the incoming data stream using phase lock loops. SyncE networks are designed according to the ITU specifications with better clocks and traceable synchronization paths rather than the usual 100ppm Ethernet clocks [14]. There are numerous PHY chips on the market that support SyncE, e.g. [6].

3. Time Semantics in Network Elements

3.1. Basic network considerations

A network bridge joins two network segments on the data link layer. Network packets are sent by the bridge based on a forwarding database that contains the MAC or IP addresses of devices connected to the bridge. Bridges introduce PDV via the following mechanisms:

- Excess network traffic during population of the forwarding address database,
- Buffering on input and output queues and variable processing time for packets,
- Successive packets taking different paths in multiply connected networks. This is common in the Internet but may also be the case in LAN environments during reconfiguration of mesh or ring topologies often used for redundancy purposes. This is obviously not present in star topologies.

Bridges are non-deterministic with respect to packet order, which depends not only on the receipt order at inputs but also on the design of the switch fabric and scheduling rules. For example, with a round-robin switch fabric, the delivery order for simultaneously received packets depends on the state of the round-robin.

PDV can also depend on the network topology. Figure 1 shows an example for a simple network. Traffic from device A to B and device B to A will not exhibit much PDV and will be independent of traffic in other parts of the network since with full-duplex and only the point-to-point traffic shown there will never be more than a single packet in a queue, provided the devices can accept all incoming traffic at line rates. By contrast the traffic from devices D and E directed at device F can expect to see PDV since two sources are feeding the same output queue in the bottom bridge. Similarly traffic between devices A and D and between C and E can experience PDV since they share a common queue on an output port of the top bridge.

Ferrari [15] summarizes efforts to bound latency and PDV for time-critical traffic other than timing

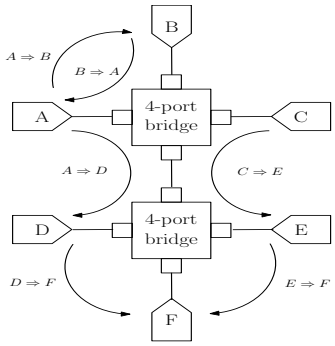

Figure 1. Simple network

messages. Much of the early work focused on ATM and wide-area communication. Ferrari makes three key observations, still true today:

- Only when network traffic is input rate limited is it possible to enforce and compute a maximum value for network latency. Admission control limits the maximum number of packets that can contend for queues thus limiting PDV. Computing this limit requires analysis of packet lengths, arrival times, and destinations — not a trivial task, see Section 3.2.
- Guaranteeing limits on latency and PDV requires all network layers and links, *including layer 2*, to enforce limits along all relevant end-to-end paths.
- Traffic must be classified as *real-time* for which latency bounds are enforced and *non-real-time* where such bounds are not enforced. Non-real-time traffic must not be permitted to degrade bounds on real-time traffic.

In recent years, the focus of network research has been primarily based on IEEE 802.1p layer 2 quality of service and the IETF IP-based DiffServ schemes. These are classification schemes based on bits in the Ethernet and IP headers respectively that allow bridges to schedule forwarding of packets. Successful enforcement of latency bounds for real-time traffic requires consistent participation by all users of the network, which is not the case today [16]. This does not bode well for using the Internet for time-critical CPS communications.

3.2. Analytical methods to compute PDV

The design and analysis of a networked CPS requires the determination of latency and PDV bounds. Software for embedded platforms is typically separated into tasks, which are assigned shared resources such as CPU cycles by a scheduler. Commonly used scheduling techniques assign priorities or deadlines to tasks to determine the order of task execution. Examples are fixed-priority scheduling such as rate-monotonic [17] or dynamic priority scheduling such as earliest deadline first [18]. Schedulability tests are performed to provide proofs that all tasks execute before their deadlines. These tests require worst case execution time (WCET) assumptions about the tasks which typically over-approximate the actual execution times leading to conservative results.

Similar techniques can be used for scheduling of messages on networks where worst case transmission times must be determined. The computational complexity increases with the complexity of the network topology since messages sent via a network from the same sender to the same receiver do not necessarily have to take the same path.

A method for analyzing performance guarantees in networks is the network calculus (NC) [19]. In network calculus, an input flow is characterized by an arrival curve α and the number of events the system can process by a services curve β. These curves specify the number of events within any time interval of length Δ. The output flow is constrained by the arrival curve $\alpha^* = \alpha \oslash \beta$, where $\oslash$ is the min-plus deconvolution [20]. Arrival and service curves can also be described in terms of the amount of resources, such as the number of processing or communication cycles, instead of number of events.

A real-time calculus (RTC) was defined for hard real-time systems using network calculus together with max-plus algebra defining an upper bound and a lower bound for the curves [21], [22]. However it is difficult to combine network calculus with real-time calculus within one homogeneous mathematical framework [21]. Furthermore, RTC cannot handle the notion of state, so some components may not be accurately modeled, e.g. when the components implement complex protocols [23]. RTC does have the advantage of scaling well, which is not the case for state-based models such as timed automata.

3.3. Hybrid Methods

Several authors have proposed combining RTC and timed automata (TA) to take advantage of both approaches [24], [25], [26] and cope with the complexity of model checking TA which is exponential in the number of clocks [27]. In [25] RTC event streams specified by arrival curves (defined in the time-interval domain) are transformed to sets of event traces specified by TA (defined in the time domain) and vice versa. The results are more accurate than the pure analytic RTC

approach, and although the computation takes longer it is still faster than with a state-based TA approach.

A different approach is presented in [24] where events are grouped in *coarse events* which are packets of real events with granularity g and the timed automaton component adapted to deal with these coarse events. The analysis can be done for different granularities thus trading off precision vs. analysis overhead.

In [26] the goal is to analyze the freshness of some data exchanged between integrated modular avionics (IMA) applications. In IMA platforms the functions share the execution and communication resources and execute in predefined time slots and communicate through an avionics full duplex switched Ethernet (AFDX) network. The network switches and processing modules are modeled by networks of timed automata. The quality of service (QoS) properties of asynchronous flows in the network are calculated with the trajectory approach [28], a technique similar to RTC for computing deterministic bounds on best and worst case traversal times (BCTT, WCTT). The network model is replaced in the automata model by a timed channel with a delay given by the interval [BCTT, WCTT].

4. Design challenges in CPS

Requirements for CPS depend highly on the application domain. However all place constraints on network latency and PDV, and most require deterministic computation and communication, i.e. given the same inputs, the system produces the same outputs. Current Ethernet network devices only do rudimentary scheduling based on best effort QoS. Given the results of the recent research described above and the presence of synchronized clocks with accuracies comparable to packet times on Ethernet, is it possible to provide better control of latency, PDV, and determinism for CPS, particularly in a LAN environment?

Figure 2 illustrates the design space we are interested in. At one extreme (E), scheduling and admission is completely uncontrolled while at the other extreme (A), admission is strictly controlled by assigning periodic access times as done in earlier versions of TTE. Later versions of TTE (B), allow three classes of service enabled by special bridges [8]. Prior work based on QoS information in Ethernet headers can provide some improvement (D), again with appropriate bridges. We propose investigating various combinations of admission control and scheduling within end devices and network bridges to provide the designer with determinacy as well as latency and PDV bounds suitable for a wider range of CPS applications.

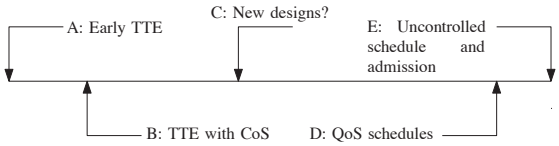

Figure 2. Design space

To cope with the complexity of the design of CPS and to allow for reasoning about a system on different levels of abstraction, model based design (MBD) is being adopted for the development of CPS. MBD allows for modeling, analyzing and evaluating system designs in different steps of the design process. Platform based design (PDB) [29] explicitly differentiates between modeling functionality and modeling architecture. PDB focuses on the integration of these models and allows for evaluation of system designs in conjunction with architectural and network properties such as time. In order to evaluate a CPS, the passage of time must be modeled.

Various tools are available for MBD and PDB. Most tools assume an unrealistic global notion of time over all platforms. In distributed CPS, every platform has its own notion of time — described by the platform clocks. An accurate model of platform clocks may also describe the clock drift, e.g. with temperature. Many modern cyber-physical systems implement clock synchronization protocols. Modeling these protocols is beneficial for evaluating their performance as well as for evaluation of network performance. The application behavior is also influenced by the network architecture and network latencies. We argue that, in order to evaluate a CPS, network components must be part of the model of the system, and timing properties introduced by the hardware must be taken into account.

We have developed an environment for describing functional aspects as well as physical properties of systems including network components in Ptolemy II [30]. This approach is explained in the following section.

4.1. Our experimental platform

Ptolemy is a modeling and simulation tool for heterogeneous systems. These systems are described as actors-oriented models. The semantics of a model, i.e. the way actors execute and communicate, is described by special model components called directors. A director defines the model of computation (MoC). Some MoC's such as discrete-event (DE) or continuous time (CT) allow for the modeling of timed systems.

Models can be composed hierarchically to form heterogeneous models that comprise more than one MoC. This environment facilitates modeling of CPS. The plant model, i.e. the physical part of the system, can be described as a continuous system, and the control laws, i.e. the cyber-part, can be represented as discrete-event systems.

To study the influence of networks on a CPS, we also represent network components as actors in the model. This requires modeling of functional as well as physical connections. In order to evaluate the influence of different network components and structures on CPS, we need an environment that allows for changing networks and physical connections with minimal changes to the functional model. Physical connections are modeled in an aspect-oriented way [31] by using concept of *quantity managers* introduced in the Metropolis project [32], [33]. Quantity managers binds physical connections with functionality models.

Figure 3 shows the Ptolemy model of the example given in Figure 1. The lines in Figure 1 represent the physical connections whereas the lines in Figure 3 show the functional connections. In the Ptolemy model, information about the physical connections between the platforms is added in form of properties on the functional connections (displayed as textual annotations in the figure). Thus, there are no lines connecting platforms and the network elements, which are represented by the actors *Bridge1* and *Bridge2* in Figure 3. The bridge functionality is implemented inside the actor.

In a simulation, a network actor receives input signals from the platform actors and sends signals to other platform actors as indicated by the functional connections. The simulation results show that network components introduce delays that can change the application behavior. The use of quantity managers facilitates change of network topologies and enables evaluation as well as static schedulability and latency analysis.

The setup described here is not trying to replace existing network simulators such as the OPNET Modeler [34] or NS-2 from the Virtual Internetwork Testbed project VINT [35]. These tools simulate networks on a much more detailed level, including the protocol stack. In Ptolemy, we can prototype new ideas for networks and evaluate those in conjunction with application models on a higher level of abstraction. We want to evaluate new implementations of network components that use additional information to allow for more deterministic communication, accurate timing of messages across networks, bounds on latencies and minimal PDV.

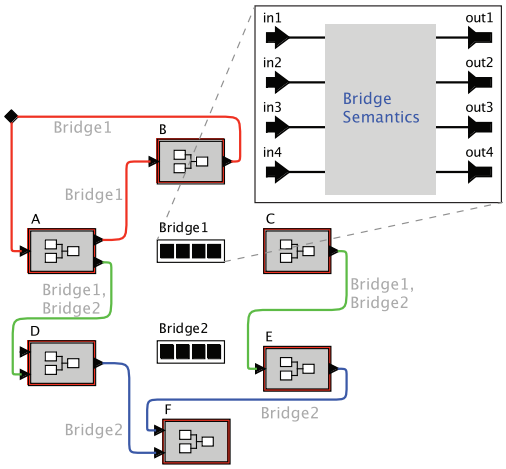

Figure 3. Modeling Networks

A realistic model of a CPS must also take into account clock drift between platforms. We consider multi-platform systems where time-synchronization protocols such as described in Section 2 provide the same notion of time on all platforms. Messages between platforms can carry timestamps acquired from the synchronized clock local to the sending platform. A useful discrete event (DE) based programming model for such a system is Ptides [36], [37]. In Ptides, event timestamps are only related to real-time at sensors and actuators. We are investigating similar relationships at network interfaces as a mechanism for enforcing admission control as well as aiding schedulability analysis. Within a platform, events have to be processed in timestamp order when they are causally related. Otherwise, events can be processed out of timestamp order. Ptides is a useful environment for simulating CPS designs but also for generating executable code that preserves a design's time semantics. This allows both simulation and experimental evaluations of designs with different combinations of platform design, admission control and bridge temporal semantics and designs.

Our initial work is centered on two multi-platform applications presenting a wide range of timing, network and other system requirements: Aircraft multi-tank fuel systems with distributed control and synchrophasors in electric power substations. The network in the first system is restricted to a local network, on board the aircraft, with requirements on the timing. The power grid application has tight requirements on the timing in the local area networks and loose requirements in the wide area network. Latencies and PDVs and their influence on the system greatly varies in these two applications. We study the effects of

network latency and PDV based on simulations and implementations of these examples.

4.2. Admission control and bridge scheduling

Admission control is a requirement for bounded latency and PDV [15]. Ptides can enforce periodic or scheduled message rates or more general bounds defined by, for example, arrival curves. Another relatively unexamined dimension of admission control is based on message temporal semantics, e.g. in order of appearance, in timestamp order, priority or class of service (CoS), or earliest deadline first (EDF).

Current bridge designs at best locally schedule messages based on CoS. Other scheduling options include strategies such as proposed in [38]. Such scheduling techniques need to be evaluated in current environments.

As illustrated in Figure 1, network topology and message connections can affect contention in bridge queues and should be considered. Likewise it is possible to have synchronized clocks in bridges which may lead to other possible scheduling options. Several of these approaches require new fields in the package header which have to be analyzed by network elements. Additional resources (time, buffer size) are also required for more elaborate scheduling techniques.

5. Conclusions and Future Work

This paper presents the main concerns when modeling and implementing networked CPS. Time and rate of communication are crucial for the correct behavior in many applications. We want to explore models for CPS that allow for expressing network delays. In the experimental setup proposed in this paper, we want to evaluate different network topologies as well as different implementations of network components. This should allow for insights into the benefits but also overheads introduced by smarter network elements such as bridges with EDF schedulers or smarter end devices that implement access control based on static or dynamic schedules. This analysis allows for the actual implementation of smarter network components.

References

[1] *Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Standard 1588-2008, 2008.

[2] (2011) Review of the markets in financial instruments directive (MiFID) MiFID II. [Online]. Available: http://www.endace.com/assets/files/announcements/20110201_Endace_MiFID_II_Comments.pdf

[3] F. J. Gonzlez-Castao, J. V. Alonso, E. Costa-Montenegro, P. Lpez-Matencio, F. Vicente-Carrasco, F. Parrado-Garca, F. Gil-Castieira, and S. Costas-Rodrguez, "Acoustic sensor planning for gunshot location in national parks: a pareto front approach." *Sensors*, vol. 9, no. 12, pp. 9493–9512, 2009.

[4] P. Moreira, J. Serrano, T. Wlostowski, P. Loschmidt, and G. Gaderer, "White Rabbit: Sub-nanosecond timing distribution over Ethernet," in *IEEE ISPCS-2009: International Symposium of Precision Clock Synchronization for Measurement, Control and Communication*, Brescia, Italy, Oct. 2009.

[5] F. Steinhauser, "IEEE 1588 for time synchronization of devices in the electric power industry," in *IEEE ISPCS-2011: International Symposium of Precision Clock Synchronization for Measurement, Control and Communication*, Portsmouth, NH USA, Sep. 2010.

[6] (2009) IEEE 1588 Synchronization over standard networks using the DP83640. [Online]. Available: http://www.national.com/an/AN/AN-1963.pdf

[7] (2011) Broadcom optimizes ethernet-based networks with end-to-end IEEE 1588 solution. [Online]. Available: http://www.broadcom.com/products/features/IEEE1588.php

[8] K. Steinhammer, P. Grillinger, A. Ademaj, and H. Kopetz, "A time-triggered ethernet (TTE) switch," in *DATE '06: Proceedings of the conference on Design, automation and test in Europe*, 2006.

[9] *Timing characteristics of primary reference clocks*, International Telecommunications Union ITU-Recommendation G.811, 1997.

[10] M. Ouellette, K. Ji, S. Liu, and H. Li. (2010) Using IEEE 1588 and boundary clocks for clock synchronization in telecom networks. [Online]. Available: http://dl.comsoc.org/livepubs/ci1/public/2011/feb/ouellette.html

[11] L. Cosart, "Packet network timing measurement and analysis using an IEEE 1588 probe and new metrics," in *IEEE ISPCS-2009: International Symposium of Precision Clock Synchronization for Measurement, Control and Communication*, Brescia, Italy, Oct. 2009.

[12] D. T. Bui, M. L. Pallec, and A. Dupas, "Packet delay variation management," in *IEEE ISPCS-2009: International Symposium of Precision Clock Synchronization for Measurement, Control and Communication*, Brescia, Italy, Oct. 2009.

[13] (2010) Timing over packet networks (ToPSync). [Online]. Available: http://www.semtech.com/timing-synchronization/timing-over-packet-networks-topsync/

[14] (2008) Synchronous ethernet: Achieving high-quality frequency distribution in ethernet ngns. [Online]. Available: http://www.cisco.com/en/US/prod/collateral/routers/ps9853/white_paper_c11-500360_ns592_Networking_Solutions_White_Paper.html

[15] D. Ferrari, "The tenet experience and the design of protocols for integrated-services internetworks," *Multimedia Syst.*, vol. 6, pp. 179–185, 1998.

[16] ——, "Humble beginnings, uncertain end: getting the internet to provide performance guarantees," in *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, 2006.

[17] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard real time environment," *Journal of the ACM*, vol. 20, no. 1, pp. 46–61, 1973.

[18] H. Chetto, M. Silly, and T. Bouchentouf, "Dynamic scheduling of real-time tasks under precedence constraints," *Real-Time Systems*, vol. 2, no. 3, pp. 181–194, 1990.

[19] R. L. Cruz, "A calculus for network delay, part i and part ii." *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 114–141, 1991.

[20] J.-Y. L. Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*. LNCS 2050: Springer Verlag, 2004.

[21] L. Thiele, S. Chakraborty, and M. Naedele, "Real-time calculus for scheduling hard real-time systems," in *Symposium on Circuits and Systems ISCAS 2000*, Geneva, Switzerland, March 2000.

[22] D. Chokshi and P. Bhaduri, "Performance analysis of flexray-based systems using real-time calculus, revisited," in *ACM Symposium on Applied Computing (SAC)*, Sierre, Switzerland, March 2010, pp. 351–356.

[23] N. Stoimenov, S. Chakraborty, and L. Thiele, "An interface algebra for estimating worst-case traversal times in component networks leveraging applications of formal methods, verification, and validation," in *Proc. 4th International Symposium on Leveraging Applications, ISoLA 2010, LNCS 6415, Springer*, Crete, Greece, Oct. 2010, pp. 198–213.

[24] K. Altisen, Y. Liu, and M. Moy, "Performance evaluation of components using a granularity-based interface between real-time calculus and timed automata," in *8th Workshop on Quantitative Aspects of Programming Languages (QAPL 2010)*, 2010.

[25] K. Lampka, S. Perathoner, and L. Thiele, "Analytic real-time analysis and timed automata: A hybrid methodology for the performance analysis of embedded real-time systems," *Design Automation for Embedded Systems, Springer Science+Business Media, LLC*, vol. 14, no. 3, pp. 193–227, 2010.

[26] M. Lauer, J. Ermont, C. Pagetti, and F. Boniol, "Analyzing end-to-end functional delays on an ima platform," *T. Margaria and B. Steffen (Eds.): ISoLA 2010, Part I, LNCS 6415*, pp. 243–257, 2010.

[27] R. Alur and D. L. Dill, "Automata for modeling real-time systems," in *Proc. of the 17th International Colloquium on Automata, Languages and Programming (ICALP'90)*, England, July 1990, pp. 322–335.

[28] J.-L. S. H. Bauer and C. Fraboul, "Applying and optimizing trajectory approach for performance evaluation of afdx avionics network," in *Proc. of the IEEE conf Emerging Technologies and Factory Automation, 2009. ETFA'09*, Mallorca, Spain, Sept 2009, pp. 1–8.

[29] A. Sangiovanni-Vincentelli, "Defining platform-based design," *EEDesign of EETimes*, 2002.

[30] J. Eker, J. W. Janneck, E. A. Lee, J. Liu, X. Liu, J. Ludvig, S. Neuendorffer, S. Sachs, and Y. Xiong, "Taming heterogeneity—the Ptolemy approach," *Proceedings of the IEEE*, vol. 91, no. 2, pp. 127–144, 2003.

[31] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. V. Lopes, J.-M. Loingtier, and J. Irwin, "Aspect-oriented programming," in *ECOOP, European Conference in Object-Oriented Programming*, vol. LNCS 1241. Finland: Springer-Verlag, 1997.

[32] F. Balarin, H. Hsieh, L. Lavagno, C. Passerone, A. L. Sangiovanni-Vincentelli, and Y. Watanabe, "Metropolis: an integrated electronic system design environment," *Computer*, vol. 36, no. 4, 2003.

[33] A. Davare, D. Densmore, T. Meyerowitz, A. Pinto, A. Sangiovanni-Vincentelli, G. Yang, and Q. Zhu, "A next-generation design framework for platform-based design," in *Design Verification Conference (DVCon)*, San Jose', California, 2007.

[34] C. Zhu, O. Yang, J. Aweya, M. Ouellette, and D. Montuno, "A comparison of active queue management algorithms using the opnet modeler," *Communications Magazine, IEEE*, vol. 40, no. 6, pp. 158 –167, Jun. 2002.

[35] V. Paxson and S. Floyd, "Why we don't know how to simulate the internet," in *Simulation Conference, 1997., Proceedings of the 1997 Winter*, Dec. 1997, pp. 1037 –1044.

[36] Y. Zhao, E. A. Lee, and J. Liu, "A programming model for time-synchronized distributed real-time systems," in *Real-Time and Embedded Technology and Applications Symposium (RTAS)*. Bellevue, WA, USA: IEEE, 2007, pp. 259 – 268.

[37] J. Eidson, E. A. Lee, S. Matic, S. A. Seshia, and J. Zou, "A time-centric model for cyber-physical applications," in *Workshop on Model Based Architecting and Construction of Embedded Systems (ACES-MB)*, 2010, pp. 21–35.

[38] J. Liebeherr, D. E. Wrege, and D. Ferrari, "Exact admission control for networks with a bounded delay service," *IEEE Transactions on Networking*, vol. 4, no. 6, pp. 885–901, 1996.