Open access • Journal Article • DOI:10.1109/JPROC.2005.849721

Ethernet-Based Real-Time and Industrial Communications — Source link

Jean-Dominique Decotignie

Published on: 31 May 2005

Topics: Industrial Ethernet, Ethernet, OSI model, Automation and Local area network

Ethernet-Based Real-Time and Industrial Communications

JEAN-DOMINIQUE DECOTIGNIE, FELLOW, IEEE

Invited Paper

Despite early attempts to use Ethernet in the industrial context, only recently has it attracted a lot of attention as a support for industrial communication. A number of vendors are offering industrial communication products based on Ethernet and TCP/IP as a means to interconnect field devices to the first level of automation. Others restrict their offer to communication between automation devices such as programmable logic controllers and provide integration means to existing fieldbuses. This paper first details the requirements that an industrial network has to fulfill. It then shows how Ethernet has been enhanced to comply with the real-time requirements in particular in the industrial context. Finally, we show how the requirements that cannot be fulfilled at layer 2 of the OSI model can be addressed in the higher layers adding functionality to existing standard protocols.

Keywords*—*Ethernet, fieldbuses, industrial communications, real time.

I. INTRODUCTION

Ethernet is now the dominant local area networking solution in the home and office environment. It is fast, easy to install and the interface ICs are cheap. Most computerized equipments now come with a built-in Ethernet interface. These are some of the reasons why a number of manufacturers of industrial control systems are now advocating the use of Ethernet, with various modifications, to support real-time communications at the factory floor. These proposals are often referred as "Industrial Ethernet" even if they correspond to different and most of the time incompatible solutions. The objective of this paper is first to explain why Ethernet is not considered as a suitable solution to support industrial communications and then to review and analyze the different additions to Ethernet to achieve an adequate solution.

Despite early attempts to use Ethernet as a real-time communication vehicle in the factories [1], [2], practitioners were reluctant to adopt this technology because of its intrinsic nondeterminism. Let us assume that two stations are waiting to transmit a message while the medium is used by another station. As soon as that station finishes transmitting, one the two stations will succeed in transmitting its waiting message. We cannot know in advance which station because the choice is random. For this reason, it is not possible to give an upper bound to the time required to transmit a message from one station to another. It is only possible to assess the probability that this time will not exceed a given value. This random aspect was rejected because the users wanted real-time guarantees (maximum transfer delay, jitter in the transmissions, and available bandwidth). These early years has seen the development of a variety of protocols providing deterministic behavior such as Token Bus, Token Ring, and a number of fieldbuses or control networks such as WorldFIP [3], Profibus [4], P Net [5], Interbus [6], AS-Interface [7], SERCOS [8], LonWorks [9], MVB [10], MIL-STD 1553 [11], DeviceNet [12], SDS [13], or CAN [14].

Most of those solutions were developed two decades ago and are now considered as too limited when compared to the available performances (mainly in terms of throughput) of non-real-time networks such as Ethernet or ATM. The requirements have also evolved toward the transmission of a higher quantity of information. There is hence a need to upgrade the existing systems toward higher speeds and performances. This is may not be feasible in some cases due to limitations in the principles of operation (CAN, for instance), but more importantly, the cost of the new developments necessary for the upgrades allied to a relatively small market is becoming a major impediment for the manufacturers. On the other hand, Ethernet and the associated protocols IP, UDP, and TCP have become so widely used that costs of interface circuits have been drastically lowered. Furthermore, Ethernet is now available at higher and higher speeds (1 Gb/s is now currently offered and 10 Gb/s will soon be), and this trend toward increasing speeds is likely to last. All these reasons have prompted a number of vendors to offer industrial communi-

Manuscript received March 30, 2004; revised January 31, 2005.

The author is with the Swiss Center for Electronics and Microtechnology (CSEM), 2007 Neuchâtel, Switzerland (e-mail: decotignie@ieee.org.).

Digital Object Identifier 10.1109/JPROC.2005.849721

cation solutions based on off-the-shelf Ethernet ICs with various modifications of the protocols to improve predictability. As we will see later, the intent of the modifications was to provide real-time guarantees. At the same time, many users are seriously considering switching to Ethernet based solutions to support communications at the plant floor. In this paper, we review the different approaches that can be used to provide real-time guarantees (such as maximum transfer delay, jitter in the transmissions, and available bandwidth) using Ethernet and the associated protocols IP [15], UDP, and TCP [16] as a communication means. The objective is to assess the guarantees that each proposal can offer. Both soft and hard real-time guarantees will be considered. We will, however, restrict our scope to industrial communications, be they control networks [17], fieldbuses [18], or sensor and actuator buses [19]. There exist other types of environments where real-time guarantees are required. Communication of multimedia streams (radio and TV channels, for instance) and sensor networks [20] are examples of them. Most of the analysis presented here can be applied to these environments, but differences exist that may lead to divergent conclusions. Analyses of the use of Ethernet in factories have been previously published [21], [22]. They present interesting but only partial insights.

Considering the OSI model [23] as a basis, real-time guarantees result from a combined effort at all layers. Starting at the highest layer, the application layer, the interaction model plays a large role in the guarantees [24]. This may be illustrated as follows. Let us assume a client–server interaction model. A client may issue a request. The network transports the request to the server. The server responds and the client gets the response through the network. If the server takes an unbounded time to respond, even if the network offers real-time guarantees, the client will have no guarantees and the real-time guarantees provided by the network protocols will be useless. This is one of the reasons why other interaction models such as publish–subscribe or producer–consumer are advocated as better suited for real-time communications [24]. This issue will not be addressed in this paper. Neither will we deal with the presentation and the session layers, as they are most of the time absent.

The next layer is the transport layer. A number of transport layers dedicated to real-time were proposed. Examples are Express Transfer Protocol (XTP) [25], and Multistream Protocol (MSP) [26]. A survey of most of the transport protocols can be found in [27]. As TCP and UDP are the most widely used, most efforts concentrate on their improvements in relationship with the Internet Protocol (IP) [28].

At the network layer, IP is the most widely used solution. The behavior of the routers has a clear impact on the real-time guarantees that can be obtained. In particular, without special routing policy, urgent messages may be unnecessarily delayed by lower priority messages that arrived earlier and are still waiting to be forwarded.

Finally, at the data link layer, the medium access control (MAC) scheme is obviously an integral part of the guarantees. To overcome the random nature of the Ethernet access protocol, a number of solutions have been proposed. Some can coexist with regular Ethernet nodes; some reuse the same hardware but are incompatible; some are compatible but cannot offer guarantees in presence of nodes that do not implement the same modifications.

As a final word, guaranteeing time-bounded operations is also a matter of software implementation. This involves adequate implementation architectures [29], as well as proper scheduling techniques. In many cases, the bottleneck in terms of throughput is not the network but the local protocol processing capability that is not sufficient or not scheduled adequately to receive all the traffic at network speed.

The paper is organized as follows. Section II introduces the requirements that an industrial communication system must fulfill. Section III gives an overview of Ethernet from its first version to the new evolutions. It explains the limitations of the technology with regards to the requirements of industrial communication. The following section reviews the different approaches to improve the real-time behavior and compares their relative merits. We restrict our scope to the proposals that reuse existing Ethernet hardware. Section V deals with two major requirements, action synchronization and temporal consistency [30]. It shows that they cannot be satisfied without adding a new layer of protocols dealing with time synchronization. Finally, Section VI concludes the paper.

II. Background

A. Application Model

Control applications operate along two different paradigms, time triggered or event triggered [31]. Time-triggered applications work periodically. They first wait for the beginning of the period (or some offset from the beginning), sample their inputs, and compute some algorithm according to the inputs and some set point data received from computers higher in the hierarchy. They then make the results available at the outputs. Inputs and outputs correspond to sensors and actuators at the lowest level in the hierarchy. At higher levels, inputs correspond to status and completion reports from the next lower level. Outputs are set points or commands to the lower level. Acquisition and distribution applications are special cases. Acquisition applications have no outputs to the process but store the output results internally. Distribution applications have no input and compute the algorithms from stored information.

Periodicity is not mandatory but often assumed because it simplifies the algorithms. For instance, most digital control theory assumes periodicity. Furthermore, it assumes limited jitter on the period and bounded latency from input instant to output instant. Acquisition and distribution applications have similar requirements in terms of periodicity and jitter.

Event-triggered applications are activated upon the occurrence of events. An event may be the arrival of a message with a new command or a completion status or the change of an input detected by some circuitry. When an event is received, the application computes some algorithm to determine the appropriate answer. The answer is then sent as an event to another application locally or remotely. The

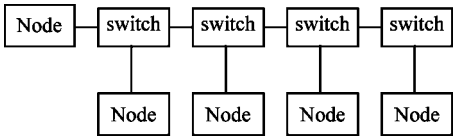

Fig. 1. Line topology based on switches.

time elapsed between the generation of the input event and the reception of the corresponding answer must be bounded. Its value is part of the requirements on the application and also the communication system if the events have to be transported through some network. Furthermore, applications should be able to assess some ordering in the event occurrences. This is usually not a problem when the event is detected and processed on the same computer. It becomes a problem when the events are detected at different locations linked by a network which may introduced some variable delay. This issue will be addressed later in the paper (Section V).

At some occasion, actions on different nodes need be synchronized. This is the case in a machine tool or a robot where the movements of the various axes must be coordinated to follow a precise path. This should be supported by the network. The use of broadcast or multicast is an option. Synchronizing distributed clocks is another (see Section V).

B. Network Model

In this paper, we will assume that all nodes are on the same subnetwork. Traffic may come from and go to external computers through routers, but this traffic has no real-time constraints. Without excluding them, we will not discuss virtual LANs as defined by IEEE 802.1Q, as they do not add to the discussion. Besides these restrictions, the network topology is free. Two extreme cases may occur. In the first one, a single node is at the root of a tree. In the second, nodes are linked to each other via three-port hubs or bridges. Each node is connected to one hub or switch. The other two ports are used to link to the adjacent hubs or bridges [32] (see Fig. 1).

C. Data Model

The network transports different kinds of data: process data, configurations and parameters, programs. Some temporal and spatial properties have to be preserved such as absolute temporal consistency, relative temporal consistency, and spatial consistency [33]. Absolute temporal consistency [34] refers to the difference in time between the current time and the time at which the information has been acquired. This is the age of information. Most data are no longer useful when they are too old. The applications should be able to decide if the information they handle is too old or not. Consider two state variables a and b. Let $[t, t_a, v_a]$ and $[t, t_b, v_b]$ be their internal representations where t_a and t_b indicate the instants at which the values v_a and v_b of a and b have been acquired. At instant t, v_a is said to be absolutely consistent if and only if

$$t - t_a \leq A_a \tag{1}$$

where A_a is the absolute consistency threshold for a.

At instant t, v_b is said to be absolutely consistent if and only if

$$t - t_b \leq A_b \tag{2}$$

where A_b is the absolute consistency threshold for b.

Relative temporal consistency applies when samples from different signals must be correlated in time. It refers to the temporal delay between the sampling instants on each signal. Two samples of two signals are said to be temporally consistent if their sampling instants differ less than a given duration, called the relative consistency threshold [30]. Many applications assume temporal consistency of their input signals. For instance, a robot controller based on the time-triggered approach will read the positions of the joints to calculate the absolute position of the extremity of a robot arm. If the positions are not sampled at the same time (they correspond the position of the joints measured at different instants), the calculation will be wrong and the controller will determine a wrong position for the arm. Networks should support relative temporal consistency by indicating when this property is present and when it is not. Using the definition of the internal representation of state variables, v_a and v_b are said to be relatively consistent if and only if

$$|t_a - t_b| \leq R \tag{3}$$

where R is the relative consistency threshold.

Spatial consistency [35] applies when the same information is copied at different locations. The replicas are said to be spatially consistent if they correspond to the same sampling instant or more generally are identical. Again, some applications assume that networks will at least tell them if consistency is present. In the robot controller example given above, the position of the arm may be used by different distributed units: one to control the position of the arm, another to prevent collisions with another robot. If, at a given time, the values are different, collision may occur. Optionally, consistency may be guaranteed.

D. Traffic Model

At least three types of traffic have to be handled: real-time periodic, real-time sporadic, and best effort. Best effort traffic corresponds to file and configuration transfers. Periodic real-time traffic is typical of time-triggered applications. It is characterized by the following parameters:

$$M_{p,i} = \{T_i, D_i, C_i\} \tag{4}$$

where T_i is the period of the transfers, D_i its relative deadline (the deadline is measured from the beginning of the period), and C_i the length of the message.

Sporadic real-time messages often result from the event-triggered applications. They can be described as

$$M_{s,i} = \{T_i, D_i, C_i\}. \tag{5}$$

T_i is the minimal interarrival time between two consecutive messages. The other parameters remain the same as in

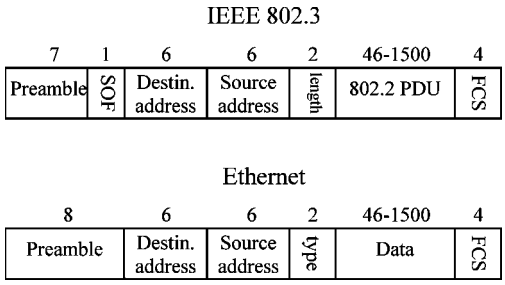

Fig. 2. Ethernet and IEEE 802.3 MAC frames (SOF: start of frame; FCS: frame check sequence).

(4). Both periodic and sporadic traffics correspond to the exchange of small messages, down to a few bytes of application data.

File transfers have a periodically sporadic pattern of arrival

$$M_{ft,i} = \{T_{i,\text{outer}}, T_{i,\text{inner}}, n_i, D_i, C_i\} \tag{6}$$

File transfers take place at most every $T_{i,\text{outer}}$. Each time they occur, n_i messages of duration C_i are sent every $T_{i,\text{inner}}$. This reflects that file transfers come at irregular intervals, but when they start, they generate a rapid flow of messages.

E. Error Model

As strange as it may be, most of the literature on real-time communications assumes an absence of errors in frame transmissions. As this paper is an overview of the approaches, we will stay with this implicit convention. The interested reader may, however, consult one of the few notable exceptions [36]–[38].

F. Soft Versus Hard Real-Time Constraints

It is usual to distinguish soft real-time constraints from hard real-time ones. The latter may not be violated, while the former may occasionally be. In the rest of this paper, we shall not make any distinction between these two categories because we will try to assess which guarantees can be obtained by the various approaches.

III. Ethernet

A. Conventional (Vintage) Ethernet

Ethernet was developed in the 1970s and emerged in products in the early 1980s. The first IEEE standard on 802.3 [39] was published in 1985. IEEE 802.3 and Ethernet have little difference. Both use the same MAC algorithm, Carrier-Sense Multiple Access With Collision Detection (CSMA/CD). The main difference is the logical link control sublayer, which is absent in IEEE 802.3 (this is covered by IEEE 802.2 [40]) and included in Ethernet. As a result, LLC protocol data units (PDUs) are transparently encapsulated in IEEE 802.3. Ethernet on its side defines a type field in the MAC frame that is used by the LLC sublayer (Fig. 2). This field is also used to carry the identification of the encapsulated protocol. The corresponding field in the IEEE 802.3 standard is the length field.

Despite the fact that compatibility between both standards was guaranteed [41], all new products are now IEEE 802.3 compliant.[1] Hence, in the rest of this paper, we will refer only to IEEE 802.3. Because, Ethernet is now used as a popular name for IEEE 802.3, we will also use the term Ethernet to designate this standard.

"Vintage Ethernet" [42] refers to the 1985 version of IEEE 802.3 and its principles of operations. In this version, all stations on a link share the same medium. In other words, every station will hear what another station emits. A station that wants to transmit first listens to the medium. If it finds the medium free for minimum duration (interframe gap), it starts transmitting. While transmitting, it monitors the medium to check for possible collisions. Collisions may occur if two or more stations start to transmit at the same time. If the frame is transmitted without collision, the process ends. If a collision occurs, the station stops transmitting after emitting a jamming sequence. The duration of this sequence has been selected to ensure that all connected stations effectively detect the collision. The station then prepares for retransmission. Instead of retransmitting immediately, a process that would likely cause another collision if all colliding stations do the same, the station randomly selects a number in the backoff range. This range is initially $0 \ldots 1$ but is doubled after each consecutive collision. In other words, if there is a collision during the first retry the interval becomes $0 \ldots 3$. If there is collision during the following retry it becomes $0 \ldots 7$. This repeats until the 10th consecutive collision after which the interval remains $0 \ldots 1023$. After 16 successive collisions, the frame transmission is aborted and an error message is returned to the higher layers. After a successful transmission, the range returns to its initial value. The randomly selected integer number in the backoff range is multiplied by the slot time to obtain the backoff delay. This is the delay the station will wait before attempting to transmit again the frame. The slot time is a system parameter calculated as the time necessary to send 512 bits (4096 bits at 1 Gbit/s). The slot time is the maximum propagation delay on the link and thus gives physical limits to the link [43].

Under light loads, the protocol yields a minimal waiting time. When the load increases, the random backoff mechanism adapts smoothly to the load of the link. As collisions are indicators of a busy medium and thus of the load, increasing the delay before retries is a good way to adjust the link load.

B. Performance Results of Conventional Ethernet

As explained in [44], obtaining accurate results concerning Ethernet is difficult because even reasonable configurations are intractable using the existing theory. The quoted paper is an interesting review of a number of theoretical and experimental analyses of Ethernet. There are a few results that have been confirmed by experiments [45]. An IEEE 802.3 network is able to operate close to 100% utilization of the bandwidth (100% utilization occurs when no time is spared in collisions and retransmission of messages lost in the collisions) when packets are long even

[1]Some "industrial Ethernet" proposals still use the Ethernet frames.

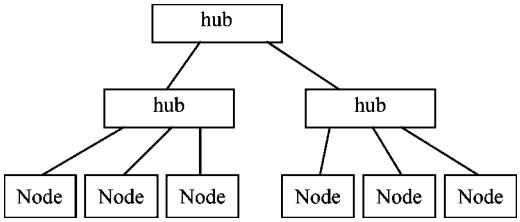

Fig. 3. Hub-based Ethernet.

with a large number of nodes. With smaller packets, the maximum utilization drops but remains much higher than the theoretical limit of 37% [46]. Note that the theoretical limit has been obtained under simplified assumptions. This explains the discrepancies with the measurements and the simulations [44]. There are some other results. The transmission delay increases nearly linearly with the size of the packet and the number of nodes. The standard deviation of the delay, which is a measure of the jitter, also increases with the number of hosts and the size of the packets but less rapidly. In summary, small packets are favorable when considering the delay and its variation. However, they lead to an inferior utilization of the network.

C. Problems With Conventional Ethernet

The MAC is usually fair and gives a minimum waiting delay when the medium is not too loaded. It is, however, impossible to guarantee a bounded message transmission time with this scheme. Under heavy loads, the 802.3 MAC may become unfair through a phenomenon called the capture effect. Let us assume that two stations A and B both have a lot of traffic to submit. At some point, their emissions collide. Station A chooses 1 as random number in the backoff range. Station B chooses 0. Station B will hence succeed at the first retry. As it has more traffic to emit, it will then send a new frame immediately. This frame will collide with station A's retry. A will double its backoff range while B starts with the initial range. It is thus likely that B will succeed again first. The same process may repeat. The result is that station B captures the medium for some period [47], [48].

This example shows that a transmission may be canceled after the maximum number of retries has been reached, even in the absence of errors on the link.

D. New Evolutions

Since 1985, IEEE 802.3 has been improved in a number of ways. In 1987, the 1 Mb/s 1BASE5 version standardized a twisted pair hub-based architecture. In this topology [43], each station is linked by a point-to-point twisted pair cable to a device called a hub that acts as a N-port repeater. The hub regenerates the signals received from one port and transmits then to all other ports. At the same time, it detects possible collisions on each port and, when one occurs, it notifies the collision on all other ports. With this new specification, the topology becomes a tree (Fig. 3). A 10-Mb/s version of the same topology followed in 1990. It was upgraded to 100 Mb/s in 1995 and 1 Gb/s in 1998. Except obvious increase in speeds, all these versions did not change the conclusion of the previous paragraph.

In 1997, project IEEE P802.3x released the standard for the so-called full duplex operations. Basically, the hub is replaced by a MAC bridge, usually called an Ethernet switch. Following IEEE 802.1D [49], the switch regenerates the information and only forwards it to the port on which the destination is attached (or a link to this destination). As any MAC bridge [50], the switch complies with the IEEE 802.3 MAC protocol when relaying the frames. If a frame is already being transmitted on the output port, the newly received frame is queued and will only be transmitted when the medium becomes idle. In addition, all cables are point to point, from one station to a switch and *vice versa* or from a switch to another. These two facts render a full duplex operation possible on each cable[2] without any collision. Hub-based solutions are limited to half-duplex because what a station transmits is sent to all other stations. By eliminating collisions, this new version was a big step toward a better predictability of the protocol. Full prediction of the transmission bounds was not yet there because overflows could still occur in the switches.

Consider two nodes A and B that transmit at full speed to a third node C. The link between A and the switch can handle the traffic. The same applies to the link between B and the switch. However, the combined traffic exceeds the capacity of the link between the switch and node C. The excess traffic accumulates in the switch until its output buffer overflows.

To avoid possible overflows in the bridge queues or in stations, a station or a bridge may send a PAUSE frame to the other side of the link. This frame notifies the other side to suspend the emission of new frames for a given duration.

Bridges forward the information based on their knowledge of the network. In particular, they learn which port must be used to reach a given node. For this, they "sniff" the received packets and look at the source address field. However, switches cannot learn to which port, or which ports, they should forward a frame sent to a multicast address. This is because there is no explicit subscription to a multicast address. By default, they must forward multicast frames to all their ports (except the ingress port). In a typical network, there may be more than a single path between any two nodes. As a consequence, a multicast frame may loop inside the network creating an unnecessary traffic that may not be marginal. To avoid loops, MAC bridges implement a spanning tree algorithm that is used to disable redundant links [49]. When the IP is used, Internet Group Management Protocol (IGMP) associations may be spied on by the switches and the resulting information used to avoid retransmitting a multicast message to ports that have no listener for that multicast group [51]. This process, often referred to as IGMP snooping, is available in many commercial switches.

E. Temporal Behavior

MAC bridges handle up to eight traffic priorities (as per IEEE 802.1D and IEEE 802.1Q), although in practice only four can be used. However, most of the time, switches have

[2]The cable has one twisted pair in each direction.

only three queues, best effort, real time, and management, per egress port. This is not enough to implement fixed priority [52] or deadline scheduling. However, this may be used to separate real-time from best effort traffic. Switches come in three flavors although only the first one is defined in the Ethernet standard.

- "Store and forward" switches receive a packet from a port. When the packet is entirely received, they check its integrity and, if it is correct, they insert the packet in the output queue linked to the selected (according to the address in the packet) output port.
- "Cut through" switches start receiving the packet. As soon as there are enough bits to check that no collision had occurred (64 B), they start inserting the packet in the queue associated to the selected output port.
- "Fast forward" switches differ from the previous ones in that they start sending the packet to the output queue as soon as there enough information to identify the output port (address fields).

Packets arriving at a given priority are queued at the egress port on a first-come, first-serve basis. When the egress link is free, the switch selects the "oldest" packet from the queue that has the highest priority among the nonempty queues. Let us assume that this is the lowest priority queue. If, just after the packet has started to be transmitted, a new packet arrives in a higher priority queue, the newly arrived packet will have to wait until the currently transmitted lower priority packet has been completely sent. There is hence a potential blocking time of the duration of the longest packet (most of the time less than 1500 B) because emission may not be preempted.

IV. Ethernet Improvements Toward Real Time

In parallel to the modifications in the standard, a number of improvements to Ethernet temporal behavior have been proposed in the literature [53]–[60]. There are three possible approaches: suppress the collisions, reduce their number, and resolve collisions in a deterministic manner. Using switches or changing the MAC is an example of the first approach. Collisions may be reduced by reservations such as in IEEE 802.11 [60] or using the Virtual CSMA approach [62]–[65] or [83]. The CAN [14] and the CSMA-DCR protocols [67] are examples of deterministic collision resolution schemes. Although this classification is commonly used, it is not very useful in practice because it does not show the degree of compatibility with Ethernet. As explained below, in this paper, we will classify the solutions according to their degree of compatibility with standard Ethernet.

In the early years, attempts have been made to modify the MAC protocol in order to improve predictability of the message transfers. This gave birth to some industrial networks such as FACTOR by APTOR [2], LAC by COMPEX, or ARLIC [1] by the SEMA group. The CSMA/DCR protocol [68] is a good example of these efforts. In case of a collision, retries are handled according to a distributed binary tree. This process guarantees a resolution in a finite number of steps. Some of these protocols have even been integrated into Ethernet chips (i.e Intel 82 596). Most of these approaches cannot be implemented without modifying the Ethernet hardware. They are technically very interesting but economically not viable today. A survey of the main solutions is presented in [69] (see also [70]). It is worth mentioning that some of them were even considered by the 802.3 committees. This is the case of the binary logarithmic arbitration method (BLAM) [71] that has been studied by the 802.3w working committee as a means to solve the capture effect. Hewlett-Packard developed 100VG-AnyLAN as a real-time Ethernet evolution. It eventually became standard as IEEE 802.12 [72].

Table 1
Classification of Solutions

Class	Behavior in presence of 802.3 compliant nodes	Examples
Non interoperable	Do not operate properly	FTT-Ethernet
Interoperable homogeneous	Operate properly but loose temporal guarantees	RETHER, traffic smoothing
Interoperable heterogeneous	Operate properly and keep temporal guarantees	EtherReal, switches + priorities

In the rest of the paper, we will only review the major improvements that use off-the-shelf IEEE 802.3 compliant hardware and thus do not require hardware alterations.[3] We thus exclude solutions that require the design of new ICs or use non-IEEE 802.3 features available only on some Ethernet ICs. Under this restriction, the modifications to the original protocol may be categorized into two classes. The first class includes all the solutions that alter the protocol in such a way that a node compliant with the new protocol cannot operate in presence of network nodes that do not implement the alterations. The second class groups all solutions that may coexist with standard products. Most of them offer guarantees under the assumption that all devices use the same modifications. They lose their advantages in presence of unmodified (IEEE 802.3 compliant) nodes. They correspond to what we shall call the homogeneous subclass. The second subclass, heterogeneous solutions, includes those proposals that offer guarantees even in presence of Ethernet nodes that do not implement the same modifications. Table 1 depicts the classification with some representative examples of solutions.

As with any taxonomy, this classification may be discussed. It has been used because it shows the degree of compatibility of the solutions. This is an important criterion when selecting a solution. Another possibility would have been to classify according to the degree of real-time guarantees. We did not follow this approach because all hard guarantees disappear in case of transmission errors [38] and only statistical guarantees may be obtained. A taxonomy around real-time guarantees would thus not give enough information.

[3]By saying this, we do not exclude modifications in firmware that are done for efficiency reasons as long as the underlying MAC remains IEEE 802.3.

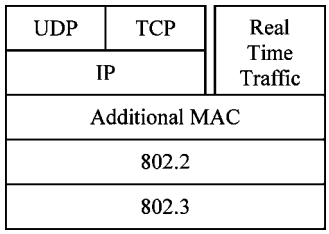

Fig. 4. Communication stack for modifications with an additional MAC.

A. Modifications That Alter Compatibility

We look here at alterations to the IEEE 802.3 MAC that make the result incompatible with standard solutions. As explained above, we assume that they can be implemented on off-the-shelf IEEE 802.3 compliant hardware.

Most of the modifications add a new MAC on top of CSMA/CD, as indicated in Fig. 4. All traffic, whether real time or not, is passed to the additional layer that controls the access to the medium. At a higher layer, real-time traffic is either handled by IP/UDP or by separate protocols.

Quite logically, all known types of deterministic MAC have been used. Examples are Time-Division Multiple Access (TDMA) [73]–[75], master–slave [76], token passing [77]–[79], slot reservation [80], or time packet release [81], [82]. At the same time, some authors [83] have suggested to reuse the MAC schemes of some fieldbuses such as PROFIBUS [4] and WorldFIP [3] on top of IEEE 802.3.

In TDMA-based systems, time is divided into slots often of equal size. Each node is given one or more slots during which it is allowed to transmit. A common sense of time is maintained either through beacons sent by a special station [84] or using a synchronized distributed clock [85]. While advocated for their robustness in fault-tolerant distributed systems [86], TDMA systems are seldom used in LANs because of their poor efficiency. For instance, it was shown that efficiency was below 4% when implementing a TDMA on gigabit Ethernet hardware [73]. This comes in particular from the delays inside the switches and the protocol software. To take into account the fact that a frame might go through a number of switches before reaching its destination, enough guard time should be left between the time slots. Furthermore, these delays diminish the accuracy of the distributed clock maintained by the participating nodes [87]. Another reason for inefficiency of the TDMA approach is related to the management of errors. Messages may be lost due to errors on the network. To be able to recover from these errors, additional time slots should be present. If each slot is statically assigned to one station, to overcome a single error per frame, the number of slots must be doubled. To reduce this overhead, some authors have introduced dynamic slot management schemes [74].

In master–slave organizations, a special station, the master, polls the other stations, the slaves. Slaves are only allowed to emit when being polled. Master–slave polling techniques are adequate as long as traffic is regular and the number of stations is not too large. If traffic is highly variable, most of the time, a station will have no traffic to send when polled. The polling overhead will grow accordingly. If the number of stations increases, the polling cycle time may become larger than the lowest message deadline (or the smallest period) and the real-time guarantees will no longer be fulfilled. It is, however, possible to poll some stations more often than others to comply with the temporal requirements. Approaches such as the cyclic executive can be used [88].

Another drawback of the polling approach is that while the master waits for an answer from the slave, no traffic may take place. On fast networks, the waiting time is dominated by the delay in the slave software. For instance, an Ethernet frame of 64 B will use the medium for 5 μs at 100 Mb/s. This is easily an order of magnitude below the time spent in the slave software. On modern Ethernet networks, the delays in the switches worsen the situation.

Token passing schemes have been largely used in real-time networks—for instance, in the IEEE 802.4 token bus [89] and the IEEE 802.5 token ring [90]. They are appreciated for their good compromise between predictability and adaptability. A station is allowed to transmit when it holds the token. When it has exhausted its right, the station transfers the token to the next station using a special message. The token is passed from station to station is a round-robin manner. The temporal properties of token bus systems have been extensively studied [91]. The IEEE 802.4 protocol is based on two main parameters, the target token rotation time and the token hold time. The token rotation time is the time elapsed between two consecutive receptions of the token at a given node. If this time is longer than the target rotation time, a node is only permitted to transmit a single message. If this time is shorter, the node may transmit its waiting traffic for a duration equal to the token hold time. By selecting different values of the hold time, different shares of bandwidth can be given to different nodes. These two parameters should be carefully calculated to ensure a proper operation. In particular, the token rotation time should be lower than the lowest message deadline. This makes token-based solutions inefficient when deadlines are very different which is often the case at sensory level. PROFIBUS [4] is based on the same principles but does not use the token hold time parameter. The IEEE 802.5 token ring keeps a single timing parameter, the token hold time. The solution is efficient but cannot be implemented on IEEE 802.3 because the physical ring is absent.

Implementing token bus protocols on top of IEEE 802.3 does not introduce any additional problem as long as switches are not used. The delays introduced by switches add an overhead to the token passing operations because no station is allowed to transmit before it has received the token. This is acceptable as long as stations have traffic to transmit, but if only one station has traffic, token passing introduces a high penalty. Finally, the main drawback on these solutions is the very long inaccessibility time in case of token loss.

Reservation techniques provide a deterministic access to all stations that can be round robin or based on priorities. All stations share a common knowledge of time (which might refer to the last sent message). Time is divided into epochs. One of the epochs is the scheduling period. This

period is further divided in to reservation slots. In the basic technique [91], each slot is assigned to a node. A node willing to transmit marks the corresponding slot. The reservations are then used to decide which station will be allowed to transmit in the transmission period that follows the scheduling period. This technique is not compatible with IEEE 802.3 silicon that cannot transmit short patterns of bits. The broadcast recognizing access method (BRAM) [92] and mini-slotted alternating priorities (MSAP) [92] are variations of this technique that can be used on regular IEEE 802.3 hardware. After the end of a transmission period, each station computes a different waiting time (which is an integer number of slots). When its waiting time has elapsed, the station senses the medium and transmits if it found it idle. As all waiting times are different, collisions are avoided. MSAP and BRAM differ in the way the waiting time is calculated. To be efficient, the reservation techniques require a very quick answer from the node. Although applicable to existing hardware, the reaction time will be long because software is involved. Slot durations must be set accordingly. Furthermore, when switches are used, the slot duration must be longer than the maximum delay incurred from one station to another across all the switches.

Some authors have tried to introduce more adaptive schemes implementing some of the protocols used in fieldbuses such as WorldFIP and PROFIBUS or adding windows for on-demand traffic on TDMA systems [93], [94]. As in FIP, FTT-Ethernet [95], [96] organizes traffic in elementary cycles of fixed size. Each cycle is divided into a synchronous window for guaranteed traffic and an asynchronous window for best effort messages. At the beginning of the elementary cycle, the master station broadcasts a synchronization message that contains the list of identifiers of messages that must be transmitted in the synchronous window of the cycle and the time at which the stations that produce them may transmit. These stations then emit the messages and all potential consumers of the messages capture them. After the synchronous window, the master polls some stations for possible asynchronous traffic. This scheme repeats after the cycle has ended. The selection of stations authorized to transmit and polled stations is based on proper traffic scheduling on the master that guarantees the real-time constraints on synchronous traffic and a fair access for the asynchronous traffic. The scheme presented in [97] may seem similar but bears important differences. The synchronous window is divided in fixed size slots. Each station that has guaranteed traffic is assigned a slot statically at startup. It decides which message to transmit in its slot according to preconfigured scheduling information. In the asynchronous window, stations may transmit freely according to the CSMA/CD protocol. At the end of the elementary cycle, there is a guard window. As soon as it begins, stations are no longer allowed to transmit except already pending traffic. In "Ethernet Powerlink" [99], a special station controls all accesses to the medium in a master–slave polling scheme. The polling order is preconfigured before runtime but may be modified later. Each elementary cycle begins with a special synchronization message from the master. Then stations are polled one after the other during the synchronous window. Each station may indicate in its response to the poll request that it has more traffic to send. The responses are sent in broadcast; traffic from node to node is hence supported. Depending on the additional traffic requests, the master will start a sporadic window in which nodes that have requested will be polled. The system offers hard guarantees for the preplanned traffic and soft guarantees for the on-demand traffic [98]. Due to the use of a central traffic manager, the master, periodicity, and precise synchronizations may be achieved (note that this is also valid for polling and TDMA systems). "Ethernet Powerlink" resembles a simplified version of WorldFIP on top of IEEE802.3. A similar solution is reported in [93].

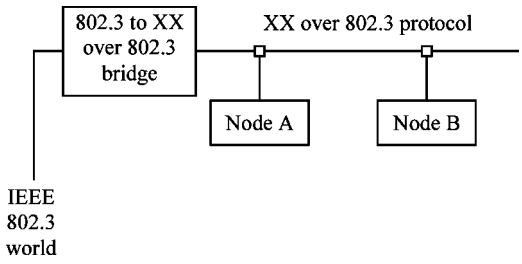

Fig. 5. Bridge isolating nodes using a MAC access protocol over IEEE 802.3 from regular IEEE 802.3 nodes.

Despite their adaptability and amenability to strict periodic operations, these solutions retain the limitations of the MAC (TDMA, master–slave, etc.) they use as explained earlier in this section. In particular, with the increase in signaling rate, the efficiency (part of the time spent using the transmission media) drops. Response times to polling are mainly due to software and have a larger and larger impact. Slot times cannot be decreased in proportion to the signaling rate increase because the transmission time uncertainties are due to software. The impact is more or less important depending on the solution. Better performances could be achieved by implementing the solutions in firmware on the board of the network interface card. However, this comes against the use of off-the-shelf interfaces. Uncertainties in the switches also add to the inefficiency.

Solutions that use another MAC protocol on top of 802.3 are not compatible with regular IEEE 802.3 protocol despite the use of off-the-shelf Ethernet hardware. First, all real-time guarantees are lost in presence of traffic from regular IEEE 802.3 nodes because these emit at will without following the rules of the other MAC schemes. Furthermore, the regular protocol management activities will declare illegal traffic coming outside the rules. For this reason, subnetworks using these protocols must be isolated from regular Ethernet subnetwork through bridges as depicted in Fig. 5.

B. Modifications That Keep Compatibility

All these improvements are compatible with IEEE 802.3 compliant devices. In other words, nodes that implement the improvements and regular Ethernet nodes will coexist and communicate. Two subclasses may be distinguished, the homogeneous and the heterogeneous one. The homogeneous subclass includes all the proposals that offer their guarantees

under the assumption that all devices implement the same improvements. Care should be taken to ensure homogeneity and interworking [50] units should be used to communicate with other networks such as the Internet. Proposals in the second subclass, heterogeneous, offer guarantees even in presence of nodes that do not implement the same modifications. They are obviously the most interesting solutions from the practical viewpoint.

1) Homogeneous Solutions: An interesting approach toward offering guarantees has been initiated by Kweon *et al.* [100]. The rationale is that most control applications work periodically and thus generate a periodic traffic. The period of transfer is such that, if occasionally some transfer does not get through, the application will not be in danger. This is for instance the case of many control applications running on programmable logic controllers (PLCs) because the period is often lower than what is required by the sampling theorem. It is thus sufficient to given statistical guarantees: probabilities that messages will be transferred within some given time bounds. The idea is that a smooth traffic, in which messages arrive at a constant rate, is less likely to suffer from collisions than a bursty traffic. The principle is borrowed from the ATM field in which it has been demonstrated that traffic smoothing improves network performances. In time-triggered applications, real-time traffic is smooth, which is adequate, but non-real-time traffic comes by bursts. At arrival and before being submitted to the link layer, this traffic must be smoothed by delaying it for longer periods. The process is the following. When a burst occurs (for instance, a download request), the messages are kept in a queue and sent one after the other at a rate which is lower than the arrival rate and acceptable by the network. Smoothing uses the token bucket principle [101]. The bucket has a maximum number of credits [the credit bucket depth (CBD)] and a replenishment period (RP). Every RP, CDB credits are added to the bucket. If the resulting number exceeds the CDB, all credits that exceed this value are discarded. When a packet is submitted, it may be sent provided there is at least one credit. A number of credits equal to the packet size is removed from the bucket. If the number of credit is zero or lower, the packet is kept until the bucket credit size becomes positive. The real-time traffic is not smoothed and is sent as soon as it arrives. The authors have shown that, provided the total load does exceed a given limit called the network wide limit [102], drop rates (rate of message transmissions that are aborted because the maximum number of retries has been exceeded) and delay variations of both the real-time and non-real-time traffics can be greatly improved. In addition, for a given drop rate, the network utilization is increased. However, the delay incurred by real-time traffic is increased by the presence of non-real-time traffic. Experiments were made on a 10-Mb/s network with a real-time traffic of 480 kb/s (5%). The delay did not exceed 1 ms in absence of other traffic. Adding a non-real-time load of 3.2 Mb/s increased the delay to 130 ms during the periods of intense traffic.

In the first approach [100], the network-wide limit was split statically to give limits for each individual station. Because this gives strong constraints at configuration time, the work has later been improved using adaptive schemes [103]–[105]. The principle is to measure the workload of the network and adapt the limit accordingly. To evaluate the network load, two means have been proposed, measuring the collisions or the throughput during a given observation window. Based on this, the station limit may be increased by a fixed amount or decreased by half, harmonic increase multiply decrease (HIMD), as in [103]. The limit may also be managed by a fuzzy controller [104] that uses both the throughput and the collision rate as a measure. This policy gives better results than the simple HIMD scheme. This in particular due to the fact that HIMD uses only collisions as a measure and that collisions occur even in absence of overload.

The traffic smoothing approach applies without any doubt to soft real-time systems. It may also be applied to hard real-time periodic (or time-triggered) systems because, as mentioned in [102], "the polling period is short enough to allow a small fraction of messages to be lost." In such a case, most systems keep the previous value while waiting for the next message.

Traffic shaping has also attracted the attention of a number of authors [106], [107]. An approach similar to Kweon's one has been proposed earlier by Sun *et al.* [107] for videoconferencing systems. Traffic is shaped statically. Packets are sent by burst of fixed size at periodic intervals. Only UDP traffic is used. The conclusions are similar in that delay variations and drop rates are reduced but delay is increased because packets have to wait for an average of half the period. Contrary to Kweon's work, the solution may be implemented easily on switch-based networks.

A second example of homogeneous solutions is given by RETHER [108]–[110]. RETHER has two operating modes, a regular CSMA/CD mode in absence of real-time traffic and a token passing mode for real-time operations. As soon as a request for real-time traffic arrives on a given node, this node initiates a transition to the token passing mode. All nodes have to acknowledge the transition and, from this point, the network operates according to the token protocol. The token circulates among two sets of nodes, the nodes that have real-time traffic (RT set) and the others. Each station of the RT set gets the token once every period. Inclusion in the RT set is based on reservation and care is taken not to exceed the available bandwidth. Each node in the RT set may have a different bandwidth requirement that translates into a different token hold time. However, all RT nodes are visited at the same period. Nodes that do not have real-time traffic are visited as soon as all RT nodes have been visited and provided there is still time available before the next token rotation period. The system goes back to CSMA/CD mode when the RT set becomes empty. The system guarantees a share of the bandwidth to nodes in the RT set. However, message deadlines should be larger than the token rotation time.

The system works on a single link (contention domain). A switch will split the system in different domains (as many domains as ports on the switch). Each domain will have an independent token operation. The results reported in [95] show that the system exhibits high overheads in case of small

packets. Software delays have a clear impact on the performances. This is in particular the case of the token management. These results are consistent with the conclusions presented earlier in this paper for token-based solutions. As long as no real-time traffic is requested, RETHER is compatible with regular IEEE 802.3 solutions. In presence of real-time traffic, RETHER becomes incompatible. For this reason, RETHER could have been included in the class of modifications that alter compatibility. We, however, wanted to emphasis the fact that it could be used in a compatible way.

2) Heterogeneous Solutions: A first example of the heterogeneous class of proposals is EtheReal [111]–[113]. EtheReal provides connection-oriented bandwidth guarantees while using regular Ethernet network adapters and drivers. The key to guarantees is the exclusive use of a properly designed switch. No other intermediate device, hub, switch, or router is allowed. With this constraint, EtheReal will provide the guarantees even in the presence of nodes that do not use EtheReal principles. The modifications on the switches can be implemented in software and do not require any special hardware. User level libraries are added to the hosts to support the establishment of connection. Finally, the solution handles rapid network reconfiguration in case of link or switch failure.

EtheReal distinguishes two classes of traffic, best effort and real time. Real-time traffic is assumed to exhibit variable bit rate (VBR) as defined in ATM.

When a station wants to transmit real-time traffic to another one, it sets a connection via its real-time communication daemon (RTCD). A connection request with the destination node IP address is sent to the first switch. The request contains the QoS parameters (average bandwidth and maximum burst size) of the connection and a 16-bit connection identifier. If QoS can be guaranteed, the switch forwards the request to the next switch on the path and the process is repeated until the last switch (to which the destination node is directly connected) is reached. The destination node is not involved in the process, as real-time connections are simplex. If the last switch accepts the QoS requirements, it returns an acknowledgment to the previous switch. The acknowledgment propagates back to the initiating node. Each switch on the path sets up a "routing" entry with the connection identifier and the QoS needs. Note that the connection identifier is different on each point-to-point link and each switch makes a translation between the ingress identifier and the output identifier. In this way, it is easier to ensure that identifiers are unique for each link. The initiating node RTCD finally creates a proxy MAC address and a proxy IP address that contain the 16-bit connection identifier. It, then, uses the Address Resolution Protocol (ARP) cache deposit mechanism to bind the proxy MAC address to the proxy IP address.

The proxy IP address is used by the initiating node application when it wants to send a real-time packet on this connection. Real-time packets are sent using UDP. The IP layer uses ARP to find the corresponding proxy MAC address. This address is recognized by the first switch that uses its internal "routing" table to find the output link and QoS parameters. It also modifies the lower 16 bits of the proxy MAC address with the output connection identifier. The process is repeated until the destination node is reached.

Best effort traffic may use TCP or UDP. Switches will forward this traffic in a usual way (as would regular Ethernet switches do) without special guarantees. Proxy IP and Ethernet addresses are chosen carefully to avoid conflicts with existing networks [112].

Ethereal also takes care of real-time multicast traffic and spanning tree rebuilds. The authors show that rebuilding the spanning tree when new links are detected, or existing links disappear, is much quicker than what is obtained with the standard 802.1D. A maximum of 32 ms is reached with 1000 switches whereas the standard specifies a limit of 30 s. This result is very interesting, as during this time some real-time traffic may not be able to go through resulting in an inaccessibility period. When multicast is used, an EtheReal switch is more efficient than a regular Ethernet switch that has to transmit a multicast message to all its output ports. Through the establishment of multicast groups [112], an EtheReal switch knows on which ports the members of the group are reachable. This is often a subset of the output ports, and network load is hence reduced.

EtheReal is a solid and complete solution that accepts "noncompliant" nodes while maintaining the guarantees. However, as it relies on local addressing schemes, its use is limited to subnetworks and the real-time traffic cannot cross routers without losing all guarantees.

EtheReal gives better results than IEEE 802.1D concerning the time necessary to rebuilt the spanning tree. This long delay in 802.1D may be a problem in pure Ethernet-based solutions and should be carefully addressed in the design of the network. In extreme cases, redundant links should be avoided or link aggregation used (see clause 43 in [43]).

The use of full duplex Ethernet and switches constitutes another class of solutions that support all types of IEEE 802.3 compliant nodes and still offer guarantees as long as all the relevant protocols are fully used.

The use of switches to offer real-time guarantees in factory communications has been suggested and analyzed by a number of authors [114]–[121], [21]. The first idea is to build a network with regular Ethernet nodes, switches, and full duplex mode. This means that each node is hooked to a single port of a switch. This architecture suppresses all collisions but does not make the solution deterministic. Assume a network with a single controller and a number of input and output nodes. All traffic comes from the controller or is addressed to the controller (Fig. 6). The application is periodic getting the input (sensor) values, computing some function and sending the results to the output nodes. At the beginning of the period, all the sensor nodes will send their input values at the same time and at full network speed. The network link between the top switch and the controller will be overloaded. If the buffer inside the switch is not deep enough to temporarily store the excess traffic, there is a risk of overflow of the buffer, with frame losses as a consequence. Using

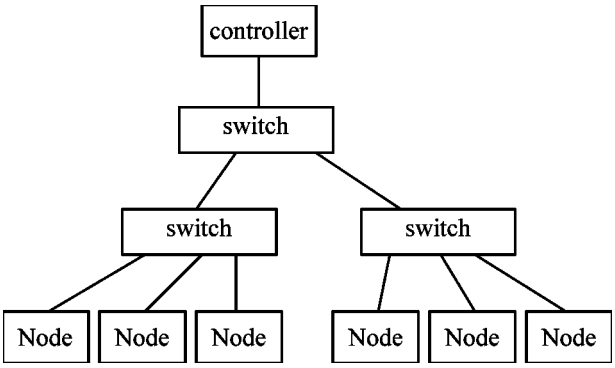

Fig. 6. Centralized application architecture using switches.

PAUSE frames may not be sufficient to avoid this overflow because the nodes that will receive PAUSE may be transmitting a long message and will only pause after that emission [118].

Under the assumption of infinite buffer length, it is possible to perform a schedulability analysis of the hard real-time traffic and determine the size of the buffers necessary for the soft real-time traffic [122]. Results are, however, statistical. A similar analysis has been made by Choi [123].

A way to reduce these showers of traffic is to use some traffic shaping. Traffic shaping will not be very effective on real-time traffic in the factories because this traffic is made of small packets. It is, however, effective to reduce the penalty introduced by best effort traffic that has typically longer packets. Analytical and experimental results show that very large link utilization may be obtained while keeping the delay below 1 ms on 100-Mb/s Ethernet [124].

The influence of best effort traffic may possibly be reduced by using the Ethernet frames priority field as defined in IEEE 802.1D (or IEEE 802.1Q). Nodes mark the outgoing frames according to the traffic priority. The best effort traffic is given the lowest priority. As described above, switches handle traffic according to the priorities. However, if a low-priority packet is being emitted on an egress port, the high-priority traffic will have to wait. This is shown clearly in the analysis done by Jasperneite *et al.* [32] using the network calculus. The assessment was conducted with four traffic classes with a traffic shaper for each class. The authors considered a "line" topology (Fig. 6) with 50 switches that closely matches the needs for cabling simplification. In the absence of prioritization, transmission delays and the variability of the transmission delays increase with the load. With prioritized handling of traffic in the switches, all messages with high priority experience a lower mean transmission delay as compared to the case without priorities. The variability of the delay, delay jitter, is also reduced. Furthermore, the delay and its variation is much less impacted by the network load. This shows that priorities may be a good mean to improve the predictability of the message transfers.

Finally, it is clear from most studies that all traffic should be controlled in order to obtain guarantees. In absence of priorities, a noncontrolled node may send a huge quantity of traffic, thus penalizing the real-time traffic. Fortunately, switches are capable of what is called traffic classification. They are able to mark all traffic coming from one port at a given priority. All traffic coming from the external world should go through such classification. The best compromise is to mark all this traffic at the lowest priority. As mentioned above, this traffic may still cause blocking in switches, but its influence is reduced.

V. Synchronization and Consistency

Up to now, the discussion has been on guaranteeing response times for temporally constrained traffic. This is very important in industrial communications but not enough to fulfill all the requirements. In this section, we describe different ways to address the requirements in terms of:

- action synchronization;
- relative temporal consistency;
- absolute temporal consistency;
- event ordering.

As explained in Section II, in many applications, actions on different nodes or sites need to be synchronized precisely. For instance, in substations for electricity distribution, five classes of synchronization accuracy have been defined [125] ranging from 1 μs in class T5 to 1 μs in class T1. In some cases, the synchronized actions must be repeated periodically with strict jitter bounds. Control of the axis movements in a robot or a machine tool is a typical example.

Relative temporal consistency is similar in its essence. It requires that input signals at different nodes should be sampled within strict temporal bounds (relative consistency threshold). Sampling at different nodes should hence be synchronized. The relative consistency threshold, or synchronization accuracy, is a small fraction of the sampling period.

Absolute temporal consistency requires the knowledge of the temporal relationship between the time of sampling (or the instant of occurrence of the event) and time of use of the sample (or the event). Event ordering is a special case is which the temporal relationship must be known between two or more events.

Support for synchronization and consistency may be based on broadcast of messages and distributed synchronized clocks. In the fieldbus domain, they have traditionally been based on broadcast of messages. Most industrial Ethernet initiatives use some form of distributed synchronized clock. Even if broadcast of messages may be considered as a very simplified form of synchronized clocks, we will present them separately.

A. Solutions Based on Broadcast Messages

Synchronized actions may be triggered using a message sent to all nodes that need to start the actions simultaneously. The simplest solution is to use a broadcast message that contains in its body some indication of the action to trigger. All nodes will receive the message and each node that recognizes the action as a possible local action will execute the action. The synchronization accuracy may be very good in a shared medium because the only sources of inaccuracies are the difference in propagation time and the variations in the receiver

message reception process and action activation. With the introduction of switches on the path, variations may be larger. As explained above, the use of priorities in the switches may reduce the variations, also called delay jitter. The variations may, however, remain quite large. The synchronization message may reach the switch while a maximum length low-priority message is already in the output port queue. With a standard maximum transmission unit (MTU) of 1500 B, the variation may reach 120 ms at 100 Mb/s. This worsens when several switches are on the path. A solution is to make sure that the queues will be empty when the message is sent. This may be done by prohibiting any traffic at some points in time, for instance, by controlling the access to the medium in a central manner. Because it is nearly constant, the unavoidable remaining delay in the switches may then be taken into account in the action synchronization.

Triggering strictly periodic actions using broadcast messages is relatively easy when the MAC is centralized or based upon some TDMA scheme. Switches must, however, be taken into account as explained above. With distributed medium access, some form of synchronized clocks must be used.

Indication of relative temporal consistency may also be based upon broadcast messages. A message is used to trigger sampling on the different nodes. This message may contain some monotonically increasing value, the stamp. It is hence sufficient to transfer this stamp together with the sample value. The receiver may compare the stamps of the different samples to assess the relative temporal consistency property of the variables [35]. The relative consistency threshold depends on the quality of synchronization of the sampling instants. It may be evaluated in the same manner the action synchronization accuracy is. Periodic sampling is also possible with the restrictions mentioned for synchronized actions.

Absolute temporal consistency may be based on local mechanism, as in WorldFIP [3], or may use the message stamp [35]. In the first case, the data is tagged with the refreshment status when transmitted. The refreshment status flag is true if the time elapsed since sampling is below a given value. It is false otherwise. The consumer elaborates the promptness status that is true if less than a given duration has elapsed since reception of the value. It is false otherwise. By checking both flags, it is possible to know whether a value is too old or not. The mechanism is called the asynchronous consistency status in WorldFIP. There is a synchronous version in which for the promptness status (and also the refreshment status) the elapsed time starts at the reception of a synchronization message. An alternative to the WorldFIP synchronous consistency is to use the value of the synchronization message (the tag) to stamp the transmitted values. As each consumer knows the time at which the synchronization message has been sent, it is able to calculate the age of the value and hence its absolute temporal consistency status [35].

The same mechanism may be used to establish the precedence and simultaneity between events. Synchronization messages are sent regularly. They carry a monotonically increasing value used as a tag. Events are then tagged with the last received value. The tag may additionally be coupled with the time elapsed since the last reception of the synchronization message. They both can be used to assess ordering between events. As for the synchronization of actions, variations in transmission of the synchronization messages limit the quality of the resulting information. The sources of variation and their amplitude should be carefully assessed. As a general rule, for a total variation of D, two events must be considered simultaneous if their stamps are less than 2 D apart.

B. Solutions Based on Synchronization of Clocks

All the requirements presented in this section (Section V) can in principle be solved using some form of synchronized distributed clocks. Actions may be synchronized by starting them at a given (preset) instant. Provided the clocks at the various nodes show the same time, the activations will be performed simultaneously. Periodic actions may use the same mechanism.

Relative temporal consistency may be supported by sampling at the occurrence of a preset instant in time. Each samples value is then transmitted with the value of the clock at the sampling instant. By comparing the sampling instants, the consumer of the values may check whether the consistency property is present or not. Similarly, the difference between the current time and the time of sampling transmitted with the value gives directly the age of the information and thus its absolute consistency status. Finally, events may be tagged with their instant of occurrence given by the local value of the clock. The couple, event and tag value (time stamp), is then transmitted. As the clocks of the various nodes are synchronized, the order of occurrence of the events may be assessed.

The accuracy of action synchronization, the capacity to establish precedence between events and the quality of the consistency status are directly related to the accuracy of synchronization between the clocks of the participating nodes. Distributed clock synchronization has been the subject of intensive studies since the 1970s. A review of the techniques is outside the scope of this paper (see [126], for instance). It is, however, worth recalling one important theoretical result concerning the achievable accuracy. In 1984, Lundelius and Lynch proved the following theorem [127]: "No clock synchronization algorithm can synchronize a system of n processes to within g, for any $g < e(1 - 1/n)$" where e is the difference between the maximum and the minimum message transfer delays. More precisely, g is the synchronization accuracy and e represents the temporal uncertainty in the transfer delay. The theorem has been demonstrated for perfect clocks, but is also valid if clocks drift apart. There are many sources of uncertainty: MAC, software latency, and operating system. To minimize their effect, part of clock synchronization must be implemented at the lowest layers. Even if this is controlled, each switch is another source of variations. Under the assumption given above (1500-B MTU, 100 Mb/s), having a single switch between two nodes will limit the worst-case synchronization accuracy to 60 ms. In practice, the average deviation between clocks is much smaller. It

is even possible to calculate the probability that the accuracy lies below a certain threshold [129]. However, this represents only a statistical guarantee and, when hard guarantees are required, the Lundelius and Lynch theorem applies.

The Network Time Protocol [128] is the most widely used clock synchronization protocol on the Internet. However, its accuracy is not adequate to reach the necessary quality of synchronization (down to a few microseconds). IEEE 1588 is a newly accepted protocol [85] that targets high accuracy synchronization. It has been designed for network measurements and to synchronize operations of control systems. To achieve submicrosecond accuracy, this protocol requires a hardware implementation [130] but looser values, in the order of 10–100 μs, may be obtained using properly designed software-only solutions. IEEE 1588 is able to take into account the uncertainty introduced by routers and switches through the concept of boundary clock. Each switch or router must implement this functionality. It has been shown that boundary clocks behave reasonably as long as switches are not cascaded [87]. At the fieldbus level, the line topology (Fig. 1) is often preferred to the tree topology. In this topology, the number of cascaded switches may become large. Jasperneite *et al.* have shown that a cascade of three switches may degrade the accuracy by an order of magnitude. With ten switches, the accuracy is even 40 times the accuracy achieved with a single switch. This degradation may be avoided with the use of "bypass clocks" in the switches [87].

Synchronization and consistency functionality requires additional protocols. Due to the stringent temporal constraints, part of the protocols has to be supported in the lower layer. This is in particular necessary to reduce uncertainties for clock synchronization. Most of the functionality may be included as an additional layer on top of the transport layer. Note that the behavior of the real-time kernel, or the operating system, may also introduce uncertainties that are prejudicial to the accuracy.

VI. Conclusion

At the time of redaction of this paper, there are nearly a dozen proposals[4] for an "industrial Ethernet" that supports real-time communications at the factory floor. Most of them are still maturing but it is likely that a number of them will coexist on the market. This paper has given the necessary background to judge and compare these proposals. The solution may come from the consumer market. With the evolution of the Internet toward higher QoS, particularly to support continuous media, voice, and images, the original 802.3 protocol has been enhanced and completed with new standards (priorities, switches, clock synchronization) that fill most of the requirements for an industrial solution. This solution has the advantage of being standard and recognized by most of the vendors. As a wireless extension with QoS is also available (IEEE802.11e), it may be the right choice.

[4]PROFINET, Ethernet Powerlink, JetSync, Ethernet/IP, SERCOS III, Modbus-TCP, Ethernet/IP, EtherCAT, PowerDNA, Real-Time Publish–Subscribe, and SynqNet.

References

[1] S. Natkin and A. Woog, "ARLIC: Une architecture de réseau local industriel," presented at the Nouvelles Architectures de Communication Conf., Paris, France, 1983.

[2] J. Boudenant *et al.*, "Lynx: An advanced deterministic CSMA-CD local area network prototype," in *Proc. Advanced Seminar on Real-Time Local Area Networks*, 1986, pp. 177–192.

[3] *General Purpose Field Communication System, vol. 3/3 (WorldFIP)*, CENELEC Std. EN 50170, 1996.

[4] *General Purpose Field Communication System, vol. 2/3 (Profibus)*, CENELEC Std. EN 50170, 1996.

[5] *General Purpose Field Communication System, vol. 1/3 (P-NET)*, CENELEC Std. EN 50170, 1996.

[6] *High Efficiency Communication Subsystem for Small Data Packages*, CENELEC Std. EN 50254, 1998.

[7] *Low-Voltage Switchgear and Controlgear—Controller-Device Interfaces (CDI's)–Part 2: Actuator Sensor Interface (AS-i)*, IEC Std. 62 026-2, 2000.

[8] *Electrical Equipment of Industrial Machines—Serial Data Link for Real-Time Communication Between Controls and Drives*, IEC Std. 61 491, 2002.

[9] *Control Network Specification*, Electron. Ind. Alliance Std. EIA-709.1, Mar. 1998.

[10] H. Kirrmann and P. Zuber, "The IEC/EEE train communication network," *IEEE Micro*, vol. 21, no. 2, pp. 81–92, Mar.–Apr. 2001.

[11] M. Haverty, "MIL-STD 1553—A standard for data communications," *Commun. Broadcasting*, vol. 10, no. 1, pp. 29–33, Jan. 1986.

[12] *Low-Voltage Switchgear and Controlgear—Controller-Device Interfaces (CDI's)—Part 5: Smart Distributed System (SDS)*, IEC Std. 62 026-5, 2000.

[13] *Low-Voltage Switchgear and Controlgear—Controller-Device Interfaces (CDI's)—Part 3: DeviceNet*, IEC Std. 62 026-3, 2000.

[14] *Road Vehicles—Exchange of Digital Information—Controller Area Network (CAN) for High-Speed Communication*, ISO Std. 11 898, 1993.

[15] "Internet Protocol," Internet Eng. Task Force (IETF), IETF RFC 791, Sep. 1, 1981.

[16] "Transport Control Protocol," Internet Eng. Task Force (IETF), IETF RFC 793, Sep. 1, 1981.

[17] F. Lian, J. Moyne, and D. Tilbury, "Network design considerations for distributed control systems," *IEEE Trans. Control Syst. Technol.*, vol. 10, no. 2, pp. 297–307, Mar. 2002.

[18] P. Pleinevaux and J.-D. Decotignie, "Time critical communication networks: Field busses," *IEEE Network*, vol. 2, no. 3, pp. 55–63, May 1988.

[19] J.-D. Decotignie and P. Pleinevaux, "A survey on industrial communication networks," *Ann. Telecommun.*, vol. 48, no. 9–10, pp. 435–448, Sep.–Oct. 1993.

[20] J. Stankovic, T. Abdelzaher, L. Chenyang, L. Sha, and J. Hou, "Real-time communication and coordination in embedded sensor networks," *Proc. IEEE*, vol. 91, no. 7, pp. 1002–1022, Jul. 2003.

[21] Z. Wang, Y. Song, J. Chen, and Y. Sun, "Real time characteristics of Ethernet and its improvement," in *Proc. 4th World Congr. Intelligent Control and Automation*, vol. 2, 2002, pp. 1311–1318.

[22] M. Alves, E. Tovar, and F. Vasques. (2000, Jan.) Ethernet goes real-time: A survey on research and technological developments. Polytech. Univ. Porto, Porto, Portugal. [Online]. Available: http://www.hurray.isep.ipp.pt

[23] *Information Processing Systems—Open Systems Interconnection—Basic Reference Model: The Basic Model*, ISO/IEC Std. 7498 : 1, 1996.

[24] J.-P. Thomesse, "Time and industrial local area networks," in *Proc. 7th Annu. Eur. Computer Conf. Computer Design, Manufacturing and Production (COMPEURO'93)*, pp. 365–374.

[25] R. Sanders and A. Weaver, "The Xpress Transfer Protocol (XTP)—A tutorial," *Comput. Commun. Rev.*, vol. 20, no. 5, pp. 65–80, Oct. 1990.

[26] T. F. La Porta and M. Schwarz, "The multistream protocol: A highly flexible high speed transport protocol," *IEEE J. Sel. Areas Commun.*, vol. 11, no. 4, pp. 519–530, May 1993.

[27] S. Iren, P. D. Amer, and P. T. Conrad, "The transport layer: Tutorial and survey," *ACM Comput. Surv.*, vol. 31, no. 4, pp. 360–404, Dec. 1999.

[28] M. El-Gendy, A. Bose, and K. Shin, "Evolution of the Internet QoS and support for soft real-time applications," *Proc. IEEE*, vol. 91, no. 7, pp. 1086–1104, Jul. 2003.

[29] F. Vamparys, "Maîtrise de la complexité d'une plate-forme de communication par une approche système (Mastering the complexity of a communication platform using a systemic approach)," Ph.D. dissertation, Swiss Federal Institute of Technology, Lausanne, 1996.
[30] H. Kopetz, "Consistency constraints in distributed real time systems," in *Proc. 8th IFAC Workshop Distributed Computer Control Systems*, 1988, pp. 29–34.
[31] ——, "Event-triggered versus time-triggered real-time systems," in *Lecture Notes in Computer Science, Operating Systems of the 90s and Beyond*. Heidelberg, Germany: Springer-Verlag, 1991, vol. 563, pp. 87–101.
[32] J. Jasperneite, P. Neumann, M. Theis, and K. Watson, "Deterministic real-time communication with switched Ethernet," in *Proc. 4th Int. Workshop Factory Communication Systems*, 2002, pp. 11–18.
[33] P. Raja and J.-D. Decotignie, "Modeling and scheduling real-time control systems with relative consistency constraints," in *Proc. 6th Euromicro Workshop Real-Time Systems*, 1994, pp. 46–52.
[34] H. Kopetz and K. Kim, "Temporal uncertainties in interactions among real-time objects," in *Proc. 9th Symp. Reliable Distributed Systems*, 1990, pp. 165–174.
[35] J.-D. Decotignie and P. Prasad, "Spatio-temporal constraints in fieldbus: Requirements and current solutions," in *Proc. 19th IFAC/IFIP Workshop Real-Time Programming*, 1994, pp. 9–14.
[36] K. Tindell, A. Burns, and A. J. Wellings, "Calculating Controller Area Network (CAN) message response times," *Control Eng. Pract.*, vol. 3, no. 8, pp. 1163–1169, Aug. 1995.
[37] N. Navet, Y.-Q. Song, and F. Simonot, "Worst-case deadline failure probability in real-time applications distributed over controller area network," *J. Syst. Architecture*, vol. 46, no. 7, pp. 607–617, Apr. 2000.
[38] I. Broster, A. Burns, and G. Rodriguez-Navas, "Comparing real-time communication under electromagnetic interference," in *Proc. 16th Euromicro Conf. Real-Time Systems*, 2004, pp. 45–52.
[39] *IEEE Standards for Local Area Networks: Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*, ANSI/IEEE Std. 802.3-1985.
[40] *Information Technology—Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks—Specific Requirements. Part 2: Logical Link Control*, ISO/IEC Std. 8802-2, ANSI/IEEE Std. 802.2, 1998.
[41] "Information processing systems—Telecommunications and information exchange between systems—Local and metropolitan area networks–Technical reports and guidelines part 5: Media access control bridging of Ethernet V2.0 in local area networks," ISO/IEC, Tech. Rep. 11 802-5 : 1997(E), 1997.
[42] B. Vouga, "How Ethernet becomes industrial," presented at the Workshop Ethernet as a Fieldbus, Geneva, Switzerland, 2001.
[43] *IEEE Standards for Local Area Networks: Part 3: Carrier Sense Multiple Access With Collision Detect on (CSMA/CD) Access Method and Physical Layer Specifications*, IEEE Std. 802.3, 2002.
[44] J. Wang and S. Keshav, "Efficient and accurate Ethernet simulation," in *Proc. Conf. Local Computer Networks*, 1999, pp. 182–191.
[45] D. Boggs, J. Mogul, and C. Kent, "Measured capacity of an Ethernet: Myths and reality," in *Proc. SIGCOMM'88*, pp. 222–234.
[46] R. Metcalf and D. Boggs, "Ethernet: Distributed packet switching for local computer networks," *Commun. ACM*, vol. 19, no. 7, pp. 395–404, Jul. 1976.
[47] K. Ramakrishnan and H. Yang, "The Ethernet capture effect: Analysis and solution," in *Proc. 19th Conf. Local Computer Networks*, 1994, pp. 228–240.
[48] B. Whetten, S. Steinberg, and D. Ferrari, "The packet starvation effect in CSMA/CD LAN's and a solution," in *Proc. 19th Conf. Local Computer Networks*, 1994, pp. 206–217.
[49] *IEEE Standards for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges*, IEEE Std. 802.1D, 1990.
[50] R. Perlman, *Interconnections—Bridges and Routers*, 2nd ed. Reading, MA: Addison Wesley, 2000.
[51] M. Christensen *et al.*, "Considerations for IGMP and MLD snooping switches," Internet Eng. Task Force (IETF), Internet Draft, May 2004.
[52] D. Katcher, S. Sathaye, and J. Strosnider, "Fixed priority scheduling with limited priority levels," *IEEE Trans. Computers*, vol. 44, no. 9, pp. 1140–1144, Sep. 1995.
[53] O. Ulusoy, "Network access protocols for hard real-time communication systems," *Comput. Commun.*, vol. 18, no. 12, pp. 943–948, Dec. 1995.
[54] I. Chlamtac, "An Ethernet compatible protocol for real-time voice/data integration," *Comput. Netw. ISDN Syst.*, vol. 10, no. 2, pp. 81–96, Sep. 1985.
[55] R. Court, "Real-time Ethernet," *Comput. Commun.*, vol. 15, no. 3, pp. 198–201, Apr. 1992.
[56] S. Norden, S. Balaji, G. Manimaran, and C. Siva Ram Murthy, "Deterministic protocols for real-time communication in multiple access networks," *Comput. Commun.*, vol. 22, no. 2, pp. 128–136, Jan. 1999.
[57] O. Sharon and M. Spratt, "A CSMA/CD compatible MAC for real-time transmissions based on varying collision intervals," *Comput. Netw.*, vol. 35, no. 2–3, pp. 117–142, Feb. 2001.
[58] S. Norden, G. Mamiran, and C. Siva Ram Murthy, "New protocols for hard real-time communication in the switched LAN environment," in *Proc. 23rd Annu. Conf. Local Computer Networks (LCN '98)*, pp. 364–373.
[59] K. Chen, "A versatile access scheduling scheme for real-time local area networks," in *Proc. IEEE INFOCOM '92*, vol. 2, pp. 804–810.
[60] V. Gupta, "A distributed backoff algorithm to support real-time traffic on Ethernet," *ACM Oper. Syst. Rev.*, vol. 35, no. 3, pp. 43–66, Jul. 2001.
[61] *Information Technology—Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks—Specific Requirements. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, ISO/IEC Std. 8802-11; ANSI/IEEE Std. 802.11, 1999.
[62] M. Molle and L. Kleinrock, "Virtual time CSMA: Why two clocks are better than one," *IEEE Trans. Commun.*, vol. 33, no. 9, pp. 919–933, Sep. 1985.
[63] M. Molle, "Prioritized-virtual-time CSMA: Head-of-the-line priority classes without added overhead," *IEEE Trans. Commun.*, vol. 39, no. 6, pp. 915–927, Jun. 1991.
[64] W. Kim and J. Srivastava, "New virtual time CSMA/CD protocols for real-time communication," in *Proc. IEEE Conf. Communication Software (TRICOMM '91)*, pp. 11–22.
[65] W. Zhao and K. Ramamritham, "Virtual time CSMA protocols for hard real-time communication," *IEEE Trans. Softw. Eng.*, vol. 13, no. 8, pp. 938–952, Aug. 1987.
[66] J. Kaiser, M. Livani, and W. Jia, "Predictability of message transfer in CSMA-networks," in *Proc. 4th Int. Conf. Algorithms and Architectures for Parallel Processing*, 2000, pp. 97–102.
[67] G. Le Lann, "A deterministic multiple CSMA-CD protocol," INRIA-Project Score, Internal Rep. PRO-1-002, Jan. 1983.
[68] G. Le Lann and N. Rivierre, "Real-time communications over broadcast networks: The CSMA-DCR and the DOD-CSMA-CD protocols," INRIA, Res. Rep. 1863, 1993.
[69] J. Kurose, M. Schwartz, and Y. Yemini, "Multiple access protocols and time-constrained communication," *ACM Comput. Surv.*, vol. 16, no. 1, pp. 43–70, Mar. 1984.
[70] *Proc. Advanced Seminar on Real-Time Local Area Networks*, G. Le Lann, Ed., Bandol, France, Apr. 16–18, 1986.
[71] K. Christensen, "A simulation study of enhanced arbitration methods for improving Ethernet performance," *Comput. Commun.*, vol. 21, no. 1, pp. 24–36, Feb. 1998.
[72] *Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 12: Demand-Priority Access Method, Physical Layer and Repeater Specifications*, ANSI/IEEE Std. 802.12, 1998.
[73] M. Schwarz, "Implementation of a TTP/C cluster based on commercial gigabit Ethernet components," Diploma thesis, Vienna University of Technology, Vienna, Austria, Dec. 2002.
[74] J. Lee and S. Park, "An error control scheme for Ethernet-based real-time communication," in *Proc. 3rd Int. Workshop Real-Time Computing Systems and Applications (RTCSA)*, 1996, pp. 214–219.
[75] J. Lee, "Design of a communication system capable of supporting real-time RPC," in *Proc. 2nd IEEE Int. Conf. Engineering of Complex Computer Systems*, 1996, pp. 99–102.
[76] Z. Wang, G. Xiong, J. Luo, M. Lai, and W. Zhou, "A hard real-time communication control protocol based on the Ethernet," presented at the 7th Australasian Conf. Parallel and Real-Time Systems, Sydney, Australia, Nov. 29–30, 2000.
[77] T. Chen, J. Huang, and Y. Chen, "Design and implementation of a client–server based real-time protocol on high speed Ethernet networks," in *Proc. Int. Conf. Consumer Electronics*, 1998, pp. 28–29.

[78] F. Hanssen, P. Hartel, T. Hattink, P. Jansen, J. Scholten, and J. Wijnberg, "A real-time Ethernet network at home," in *Proc. Work in Progress Session, ECRTS*, 2002, pp. 5–8.
[79] M. Shieh, J. Sheu, and W. Chen, "Decentralized token-CSMA/CD protocol for integrated voice/data LANs," *Comput. Commun.*, vol. 14, no. 4, pp. 223–234, May 1991.
[80] L. Kleinrock and M. Scholl, "Packet switching in radio channels: New conflict free multiple access schemes," *IEEE Trans. Commun.*, vol. 28, no. 7, pp. 1015–1029, Jul. 1980.
[81] D. Pritty, J. Malone, J. Smeed, D. Banerjee, and N. Lawrie, "A real-time upgrade for Ethernet based factory networking," in *Proc. Int. Conf. Industrial Electronics (IECON)*, vol. 2, 1995, pp. 1631–1637.
[82] D. Pritty, J. Smeed, and N. Lawrie, "A new class of high speed LAN access protocols based on the principle of timed packet release," in *Proc. 20th Conf. Local Computer Networks*, 1995, pp. 443–452.
[83] S. Vitturi, "On the use of Ethernet at low level of factory communication systems," *Comput. Stand. Interfaces*, vol. 23, no. 4, pp. 267–278, Sep. 2001.
[84] P. Pedreiras, L. Almeida, and P. Gai, "The FTT-Ethernet protocol: Merging flexibility, timeliness and efficiency," in *Proc. 14th Euromicro Conf. Real-Time Systems*, 2002, pp. 134–142.
[85] *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Std. 1588-2002, 2002.
[86] K. Kim and C. Serro, "Robustness of real-time local area network protocols," *Comput. Commun.*, vol. 20, no. 3, pp. 169–176, May 1997.
[87] J. Jasperneite, K. Shehab, and K. Weber, "Enhancements to the Time Synchronization Standard IEEE-1588 for a system of cascaded bridges," in *Proc. 5th IEEE Workshop Factory Communication Systems*, 2004, pp. 239–244.
[88] T. Baker and A. Shaw, "The cyclic executive model and Ada," *Real-Time Syst.*, vol. 1, no. 1, pp. 7–25, Jun. 1989.
[89] *Information Processing Systems—Local Area Networks—Part 4: Token-Passing Bus Access Method and Physical Layer Specifications*, ISO/IEC Std. 8802-4; ANSI/IEEE Std. 802.4-1990.
[90] *Information Technology—Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks—Specific Requirements. Part 5: Token Ring Access Method and Physical Layer Specifications*, ISO/IEC Std. 8802-5, ANSI/IEEE Std. 802.5-1995, 1995.
[91] N. Malcolm and W. Zhao, "The timed-token protocol for real-time communications," *IEEE Computer*, vol. 27, no. 1, pp. 35–41, Jan. 1994.
[92] I. Chlamtac, W. Franta, and K. Levin, "BRAM: The Broadcast Recognizing Access Method," *IEEE Trans. Commun.*, vol. 27, no. 8, pp. 1183–1190, Aug. 1979.
[93] D. Kim, Y. Doh, and Y. Lee, "Table driven proportional access based real-time Ethernet for safety-critical real-time systems," in *Proc. 2001 Pacific Rim Int. Symp. Dependable Computing*, pp. 356–363.
[94] W. Chen and C. Lu, "CSMA/CD/TDMA: A dynamic combination for voice and data integration," in *Proc. IEEE INFOCOM '90*, vol. 3, pp. 842–849.
[95] P. Pedreiras, P. Gai, G. Buttazzo, and L. Almeida, "FTT-Ethernet: A platform to implement the elastic task model over message streams," in *Proc. 4th IEEE Workshop Factory Communication Systems*, 2002, pp. 225–231.
[96] P. Pedreiras, L. Almeida, and P. Gai, "The FTT-Ethernet protocol: Merging flexibility, timeliness and efficiency," in *Proc. EUROMICRO Conf. Real-Time Systems*, 2002, pp. 134–142.
[97] A. Bonaccorsi, L. Lo Bello, O. Mirabella, A. Pöschmann, and P. Neumann, "A distributed approach to achieve predictable Ethernet access control in industrial," presented at the 5th IFAC Int. Conf. Fieldbus Systems and Their Applications, Aveiro, Portugal, 2003.
[98] L. Almeida *et al.*, "Schedulability analysis of real-time traffic in WorldFIP networks: An integrated approach," *IEEE Trans. Ind. Electron.*, vol. 49, no. 5, pp. 1165–1174, Oct. 2002.
[99] Ethernet Powerlink Standardization Group. Ethernet powerlink data transport services. [Online] . Available: http://www.ethernet-powerlink.org
[100] S. Kweon, K. Shin, and Q. Zheng, "Statistical real-time communication over Ethernet for manufacturing automation systems," in *Proc. 5th Real-Time Technology and Applications Symp.*, 1999, pp. 192–202.
[101] J. Turner, "Leaky bucket," *IEEE Softw.*, vol. 2, no. 3, pp. 49–61, Mar. 1985.
[102] S. Kweon and K. G. Shin, "Statistical real-time communication over Ethernet," *IEEE Trans. Parallel Distrib. Syst.*, vol. 14, no. 3, pp. 322–335, Mar. 2003.
[103] S. Kweon, K. Shin, and G. Workman, "Achieving real-time communication over Ethernet with adaptive traffic smoothing," in *Proc. 6th Real-Time Technology and Application Symp.*, 2000, pp. 90–100.
[104] A. Carpenzano, R. Caponetto, L. Lo Bello, and O. Mirabella, "Fuzzy traffic smoothing: An approach for real-time communication over Ethernet networks," in *Proc. 4th IEEE Int. Workshop Factory Communication Systems*, 2002, pp. 241–248.
[105] L. Lo Bello, M. Lorefice, M. Mirabella, and S. Oliveri, "Performances analysis of Ethernet networks in the process control," in *Proc. 2000 IEEE Int. Symp. Industrial Electronics*, vol. 2, pp. 655–660.
[106] M. Iwasaki, T. Takeuchi, M. Nakahara, and T. Nakano, "Isochronous scheduling and its application to traffic control," in *Proc. 19th IEEE Real Time-Systems Symp.*, 1998, pp. 14–25.
[107] Y. S. Sun, K. Chin-Fu, P. Yu-Chun, W. Chia-Hui, and H. Jan-Ming, "Performance analysis of application-level traffic shaping in a real-time multimedia conferencing system on Ethernets," in *Proc. 21st IEEE Conf. Local Computer Networks*, 1996, pp. 433–442.
[108] C. Venkatrami and T. Chiueh, "Design and implementation of a real-time switch for segmented Ethernets," in *Proc. 1997 Int. Conf. Network Protocols*, pp. 152–161.
[109] ——, "Supporting real-time traffic on Ethernet," in *Proc. Real-Time Systems Symp.*, Dec. 7–9, 1994, pp. 282–286.
[110] ——, "Design, implementation, and evaluation of a software-based real-time Ethernet protocol," *ACM Comput. Commun. Rev.*, vol. 25, no. 4, pp. 27–37, Oct. 1995.
[111] S. Varadarajan and T. Chiueh, "EtheReal: A host-transparent real-time Fast Ethernet switch," in *Proc. 6th Int. Conf. Network Protocols*, 1998, pp. 12–21.
[112] S. Varadarajan, "Experiences with EtheReal: A fault-tolerant real-time Ethernet switch," in *Proc. 8th IEEE Int. Conf. Emerging Technologies and Factory Automation*, vol. 1, 2001, pp. 183–194.
[113] S. Varadarajan and T. Chiueh, "Automatic fault detection and recovery in real time switched Ethernet networks," in *Proc. IEEE INFOCOM '99*, vol. 1, pp. 161–169.
[114] T. Jinggang-Wang and B. Ravindran, "BPA: A fast packet scheduling algorithm for real-time switched Ethernet networks," in *Proc. Int. Conf. Parallel Processing*, 2002, pp. 159–166.
[115] X. Fan, Z. Wang, and Y. Sun, "How to guarantee factory communication with switched Ethernet: Survey of its emerging technology," in *Proc. 28th Annu. Conf. Industrial Electronics (IECON)*, vol. 3, 2002, pp. 2525–2530.
[116] J. Chen, Z. Wang, and Y. Sun, "Switch real-time industrial Ethernet with mixed scheduling policy," in *Proc. 28th Annu. Conf. Industrial Electronics (IECON)*, vol. 3, 2002, pp. 2317–2321.
[117] ——, "Real-time capability analysis for switch industrial Ethernet traffic priority-based," in *Proc. 2002 Int. Conf. Control Applications*, vol. 1, pp. 525–529.
[118] E. Vonnahme, S. Ruping, and U. Ruckert, "Measurements in switched Ethernet networks used for automation systems," in *Proc. IEEE Int. Workshop Factory Communication Systems*, 2000, pp. 231–238.
[119] K. Lee and S. Lee, "Performance evaluation of switched Ethernet for real-time industrial communications," *Comput. Stand. Interfaces*, vol. 24, no. 5, pp. 411–423, Nov. 2002.
[120] E. Jaspemeite and P. Neumann, "Switched Ethernet for factory communication," in *Proc. IEEE Int. Conf. Emerging Technologies and Factory Automation*, vol. 1, 2001, pp. 205–212.
[121] P. Pedreiras, R. Leite, and L. Almeida, "Characterizing the real-time behavior of prioritized switched Ethernet," presented at the 2nd Int. Workshop Real-Time LAN's in the Internet Age 2003, Vienna, Austria.
[122] Y. Song, A. Koubaa, and F. Simonot, "Switched Ethernet for real-time industrial communication: Modeling and message buffering delay evaluation," in *Proc. IEEE Int. Workshop Factory Communication Systems*, 2002, pp. 27–35.
[123] B. Choi, S. Sejun, N. Birch, and J. Huang, "Probabilistic approach to switched Ethernet for real-time control applications," in *Proc. Real-Time Computing Systems and Applications Conf.*, 2000, pp. 384–388.
[124] J. Loeser and H. Haertig, "Low latency hard real-time communication over switched Ethernet," in *Proc. 16th Euromicro Conf. Real-Time Systems*, 2004, pp. 13–22.

[125] T. Skeie, S. Johannessen, and O. Holmeide, "Highly accurate time synchronization over switched Ethernet," in *Proc. 8th IEEE Int. Conf. Emerging Technologies and Factory Automation*, 2001, pp. 195–204.

[126] J. He, Z. Mammeri, and J.-P. Thomesse, "Clock synchronization in real-time distributed systems based on FIP field bus," in *Proc. 2nd IEEE Workshop Future Trends of Distributed Computing Systems*, 1990, pp. 135–141.

[127] J. Lundelius and N. Lynch, "An upper and lower bound for clock synchronization," *Inf. Control*, vol. 62, no. 2/3, pp. 190–204, Aug.–Sep. 1984.

[128] D. Mills, "Internet time synchronization: The network time protocol," *IEEE Trans. Commun.*, vol. 39, no. 10, pp. 1482–1493, Oct. 1991.

[129] K. Arvind, "Probabilistic clock synchronization in distributed systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 5, no. 5, pp. 474–487, May 1994.

[130] J. Edison and L. Kang, "Sharing a common sense of time," *IEEE Instrum. Meas. Mag.*, vol. 6, no. 1, pp. 26–32, Mar. 2003.

[131] *Digital Data Communications for Measurement and Control—Fieldbus Standard for Use in Industrial Control Systems. Parts 1 to 6*, IEC Std. 61 158, 2003.

[132] *Digital Data Communications for Measurement and Control—Part 1: Profile Sets for Continuous and Discrete Manufacturing Relative to Fieldbus Use in Industrial Control Systems*, IEC Std. 61784-1, 2003.

[133] N. Malcolm and W. Zhao, "Hard real-time communication in multiple access networks," *J. Real-Time Syst.*, vol. 8, no. 1, pp. 35–77, Jan. 1995.

Jean-Dominique Decotignie (Fellow, IEEE) received the B.S. degree in electrical engineering and the Ph.D. degree from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in 1977 and 1982, respectively.

He is head of the real-time and networking group at the Swiss Center for Electronics and Microtechnology (CSEM), Neuchatel, Switzerland. He is also a Professor at the School of Computer and Communication Sciences, EFPL. He has been very active in the fieldbus domain since its beginning. He has authored or coauthored more than 100 papers in the field and created the first and only international conference series dedicated to industrial communications. His current research interests include industrial and real-time local area networks, wireless sensor networks, distributed control systems and software engineering for real-time systems.

Dr. Decotignie is a Member of the Association for Computing Machinery and the Swiss Computer Society (SI). He has served as chairman of the IEEE Switzerland section and IEEE Region 8 representative at the Board of Directors. He is active within the IEEE Industrial Electronics Society.