by Jean-Dominique Decotignie

The Many Faces of Industrial Ethernet

Ethernet is by far the most widely installed local area network technology, popular in both homes and offices. It is fast, easy to deploy, and adapters are cheap. Most, if not all, computers have a built-in Ethernet interface. These are just some of the reasons why an increasing number of industrial automation vendors militate for the use of Ethernet—as is or with modifications—to support communications on the shop floor. They've made a number of proposals under the generic name "industrial Ethernet," all aimed at the specificities of communications in the industrial environment [1]. Most of the time, however, these proposals are incompatible. Limiting ourselves to the temporal aspects, we look at why Ethernet is generally considered inadequate as a support for computer communications on the factory floor. We also examine possible modifications to obtain an adequate solution based on Ethernet and the proposals that have been recently standardized under IEC 61158 and IEC 61784 [Ethernet is also used in other industrial application such as energy substations (IEC 61850)].

Why Consider Ethernet as a Potential Solution Now?

The idea of using Ethernet to carry industrial communications is not a novelty [2], [3]. In the 1980s, however, the idea faced strong reluctance from users because Ethernet is not deterministic. Consider what happens when two stations are waiting to transmit a message while a third station is transmitting. As soon as this third station has finished its transmission, one of the other stations will succeed in transmitting its waiting message. Yet, it is impossible to predict which of the two waiting stations will succeed first because the choice is random. Thus, it is also impossible to calculate an upper bound for the waiting time before a message is successfully transmitted from one station to another. It is only possible to derive the probability that this waiting time will exceed a given value.

This randomness caused Ethernet to be rejected in industrial environments because users demanded real-time guarantees, such as a maximum transfer time, a jitter not exceeding some threshold, or some guaranteed bandwidth. The result was the development of a variety of protocols offering deterministic behavior and a pallet of solutions under the name "fieldbus," such as WorldFIP [4], Profibus [4], P-Net [4], Interbus [5], AS-Interface [6], SERCOS [7], LonWorks [8], MVB [9], MIL-STD-1553 [10], DeviceNet [11], SDS [12], and CAN [13].

Most of these industrial communication solutions were created in the 1980s; some were first national standards in France or Germany, such as WorldFIP and Profibus, respectively, as early as 1989. (An excellent survey on the subject is found in [40]). They are now considered obsolete when their performances are compared with the results (in terms of throughput) obtained with modern networks, such as ATM or Ethernet. There has also been an evolution in terms of needs. Industrial and automotive networks now carry more and more information, higher data volumes, images, voice, and even video streams. Hence, existing solutions must be upgraded by increasing the data transmission rate and by improving performance in general.

This is sometimes impossible due to the design limitations of existing protocols; CAN is an example of such a case. The main obstacles when it comes to upgrades are economic. The high development costs required to improve performance cannot be justified for such a relatively small market. In the meantime, Ethernet and all the protocols linked to Ethernet, such as Internet protocol (IP), user datagram protocol (UDP), and transmission control protocol (TCP), are meeting with such success that network interface costs have dropped sharply. In addition, Ethernet is available at always increasing speeds. Gigabit speeds are now commonplace, and 10 Gb/s interface cards will soon be customary. This trend does not show any sign of slowing down. For all these reasons, a number of vendors have been motivated to offer industrial communication solutions based on Ethernet interfaces with various modifications and additions in terms of protocols to improve predictability and provide real-time guarantees. Simultaneously, numerous users are seriously considering solutions based on Ethernet to support their communications needs in their factories and shops. After all, Ethernet seems to work perfectly in their offices. Why wouldn't it work well in an industrial environment?

Obtaining Real-Time Guarantees

Let us take the open systems interconnection (OSI) layered model [14] as a

Digital Object Identifier 10.1109/MIE.2009.932171

1932-4529/09/$25.00©2009IEEE

basis. Obtaining temporal guarantees is not only the job of a single layer but the result of a collective effort by all the layers. Starting from the top layer—the application layer—the interaction model plays an important role [15]. Consider, as an example, a client-server relationship. A client issues a request. The network transports the request to the server. The server responds, and the network conveys the response to the client. Even if the network provides temporal guarantees, it will be impossible to provide guarantees to the client if there is no temporal bound between the reception of the request at the server and the response of the server. The network guarantees will thus be useless. This is why alternate models were created. The publish-subscribe model, for example, is much more suited to real-time communications [15]. Despite its importance, we will not address this topic here. We will also skip the presentation and session layers (that come below the application level in the OSI layered model), which are absent in the proposals. (The presentation layer is often implicit.)

The next layer, then, is the transport layer. A few transport protocols are aimed at providing real-time guarantees. EXpress transfer protocol (XTP) [16] and MultiStream protocol (MSP) [17] are representative examples. (A review of most of the transport protocols is found in [18].) Nowadays, TCP and UDP are so widely used that most of the effort is dedicated to adequate tuning of TCP parameters or to the design of real-time protocols that are compatible with TCP.

At the network layer, IP is the dominant solution. The behavior of the routers has a clear impact on the temporal bounds that may be guaranteed. Without special policy, low-priority messages that arrive earlier at a router and are waiting to be relayed may unnecessarily delay messages of higher priority. This aspect will not be addressed here because most of the proposals limit the real-time traffic to a single link, thus avoiding the transit of real-time messages through routers.

In the data link layer, the medium access policy [medium access control (MAC)] is an integral part of the guarantee chain. To alleviate the random nature of Ethernet, a number of modifications have been proposed. Some of these modifications are able to coexist with unmodified Ethernet nodes; some reuse existing network interface cards while being incompatible with regular Ethernet nodes. A last category preserves compatibility but can provide real-time guarantees only in the absence of "foreign" (unmodified Ethernet) nodes.

It is worth noting that the way the communication software is implemented plays a nonmarginal role in the guarantees. An adequate implementation architecture [19] and a good scheduler are two key aspects. In many cases, the bottleneck is not in the network itself but in the limitations of the network interface cards (NICs) and the processors when it comes to handling messages on time. Most NICs, for instance, are unable to withstand a continuous flow of messages at 1 Gb/s or even at 100 Mb/s.

Vintage Ethernet

Ethernet was developed during the 1970s to emerge as a product at the beginning of the following decade. The IEEE published its first 802.3 standard in 1985. There is little difference between Ethernet—the name given to the technology by its inventor, Robert Metcalf—and the IEEE 802.3 standard that derived from it; both use the same medium access control principle known as carrier sense multiple access with collision detection (CSMA/CD). The main difference is the logical link control (LLC) sublayer, which is included in Ethernet but left to another IEEE standard (IEEE 802.2). Due to this, the LLC protocol data units (PDUs) are transmitted and received in a transparent manner in IEEE 802.3. Ethernet defines a field for the packet type in the MAC frame (Figure 1). This field is also used to carry the identifier of the protocol used by the higher layer. The corresponding field carries the data length in IEEE 802.3.

An International Organization for Standardization (ISO) recommendation [20] has guaranteed compatibility between both solutions since 1997. Nowadays, all products are compliant to IEEE 802.3—but all personal computers use the Ethernet frames, while only a few protocols, such as the simple network management protocol (SNMP), use 802.2 frames. The advantages of 802.2 have been exploited by only a handful of industrial products. Ethernet is now the common name of the IEEE 802.3 standard [21], and both names are used to designate the same protocol. From here on out, I will use them interchangeably.

"Vintage Ethernet" is a nickname for the original version of the standard and its working principles. In this version, all the nodes on a given link share the same transmission channel. In other words, each node can hear what any other node emits. A node that wishes to transmit listens first. If it does not hear anything for a given duration (interframe gap), it starts transmitting a packet. While transmitting, the node listens to what is emitted on the channel. If it senses

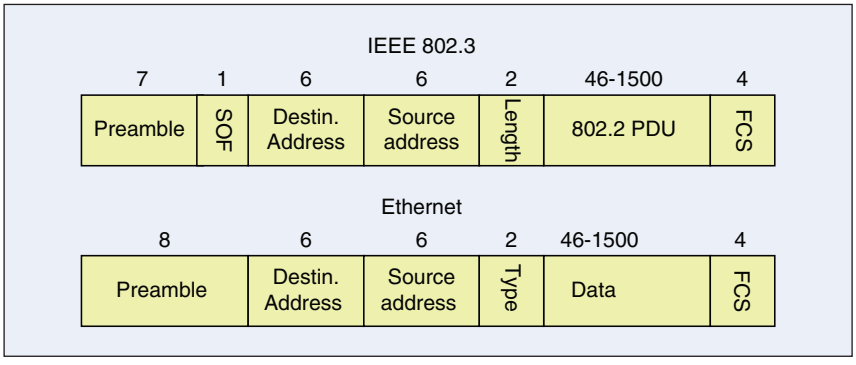

FIGURE 1 – Ethernet and 802.3 frames.

a collision—which can happen when two nodes start transmitting at the same time—it stops transmitting its data. If no collision is detected before the end of the transmission of the packet, the transmission is considered successful.

A node that detects a collision instantly stops transmitting its packet and emits a jamming sequence. The duration of this sequence has been calculated so that all connected nodes are guaranteed to detect the collision. After such a failure, the node will prepare itself to retransmit the packet. However, instead of retransmitting immediately, which would cause a second collision if all the nodes involved in the first collision adopt the same policy, the node chooses a random number in the backoff interval. This number is multiplied by a constant, the slot time, to obtain a waiting time called the "backoff duration." The node waits for this duration before attempting to retransmit the packet.

The backoff interval is initially 0…1. It is doubled after each consecutive collision. If immediately after the first collision there is a second one, the interval becomes 0…3. If a collision is detected in the retransmission, the interval is increased to 0…7 and so on, until the 10th consecutive collision, after which the interval remains the same at 0…1,023. After 16 successive attempts to retransmit, the packet transmission is cancelled, and the higher layers are informed of the failure. In such a case or after a successful transmission, the backoff interval returns to its initial value of 0…1.

The slot time is the length of time necessary to emit 512 b (4,096 b at 1 Gb/s). It corresponds to the maximum propagation time on the link (shared medium) and thus limits the span of the network and the number of repeaters.

When the network is lightly loaded, the protocol previously described exhibits a very low waiting time before sending a packet. When the load increases, the protocol smoothly adapts by gradually increasing the waiting time. Since collisions are a good indication of the link use and thus the load, increasing the delay before transmission is a very good way to adjust the load.

Vintage Ethernet Performances

It is difficult to obtain accurate theoretical results for Ethernet. Even the simple networks are quickly intractable. This is why studies of Ethernet performance make a number of simplifying assumptions (an infinite number of nodes, constant packet length, pending traffic always) and the results are merely approximations. Some of the results, however, have been confirmed by practical experiments [22].

- An IEEE 802.3 network is able to operate at close to 100% of its capacity when packets are long and there is a small number of nodes (100% capacity corresponds to the case in which no time is lost either in collisions or retransmissions).

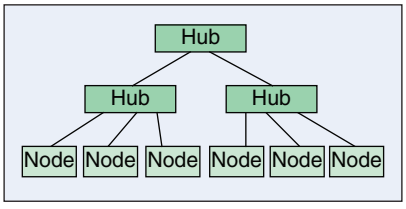

FIGURE 2 – Hub-based Ethernet topology.

- When the packets are short (the minimum is 64 B), efficiency drops but remains much higher than the "theoretical limit" of 37%. This 37% limit, widely found in the literature, has been obtained under simplifying assumptions, which explains the discrepancy with practice.
- The mean transmission delay grows quasi-linearly with the packet size and the number of participating nodes.
- The transmission delay standard deviation also increases with the number of nodes and the packet size.

To summarize, when a short transmission delay and a low jitter are preferred, it is better to use small packets. Long packets are favorable to higher network utilization.

Vintage Ethernet Problems

Ethernet MAC is ordinarily fair and the waiting time low when the network is moderately loaded. It is impossible, however, to give an upper bound to the waiting time before transmission. In addition, under high loads, Ethernet becomes unfair due to the "capture effect" [23]. Suppose two nodes, A and B, have a number of packets to transmit. At some point in time, their transmissions collide. Node A selects 1 as a random number in the backoff interval, while node B selects 0. Node B will be successful in its first retransmission. As it still has other messages, it will try immediately to transmit the next message. This transmission will collide with the emission of node A. Node A then doubles the backoff interval, whereas node B uses the initial one. Node B thus has a higher chance of transmitting successfully. This phenomenon may repeat a number of times and may allow node B to monopolize the medium for a while. In the worst case, node A may fail to transmit after 16 retries, even though the network is far from being overloaded. This is more of a theoretical problem than a practical one, however, because most NICs are unable to sustain a high throughput, and the node will therefore quickly loose the arbitration.

Evolutions

Since 1985, different aspects of IEEE 802.3 have been improved. In 1987, the 1BASE5 version standardized a solution based on twisted pairs at 1 Mb/s with a new topology. In this topology (Figure 2), each node is linked through a dedicated cable to a special element called a "hub" that acts as an N-port repeater. The hub regenerates the signal coming in at one of its ports and outputs it to all the other ports. It also detects possible collisions and reflects them on the other ports. With this new version of the standard, the bus topology is replaced by a tree-based architecture (Figure 2).

Rapidly, new versions were published. The bit rate increased to 10 Mb/s in 1990, then 100 Mb/s in 1995, and 1 Gb/s in 1998. Apart from the increase in transmission speed, the working principle of these new versions remained identical to vintage Ethernet. The conclusions drawn earlier remain valid.

Things changed drastically in 1997 with the release of the "full duplex"

Ethernet. The hub is replaced by a bridge in what's commonly called an "Ethernet switch." In conformance with the IEEE 802.1D [24] standard, a switch receives packets from all its ports but only transmits them to the port on which the destination node is connected. The switch also complies with the MAC scheme. If a packet is currently being transmitted on an output port, a new incoming packet that should be sent out on this port will be queued until the medium at the output port becomes idle. As in the previous version, cables support point-to-point links between a node and a switch (and vice versa) or between two switches. Since there is a pair of wires for each direction of transfer (one transmitter and one receiver per pair) and because the hubs are replaced by switches, the links may be exploited in full duplex without possible collisions. Note that solutions based on hubs were restricted to half-duplex operations. This new solution is much less favorable to industrial use. The most annoying features are the restricted topology and the fact that it's impossible to observe all the traffic from a single point.

Removing collisions is a great step toward better predictability of the protocol. Calculating worst-case transmission bounds is still not possible, however, because overflows may still occur in the switches and induce packet losses. Consider, for instance, two nodes, A and B, that transmit at full speed toward a third node, C (Figure 3). The link between node A and switch 1 is able to withstand the traffic. The same can be said for the link between node B and switch 1. However, the combined traffic exceeds the capacity of the link between switch 1 and node C. Packets coming from node A or node B and waiting for relay to node C pile up in switch 1 until the waiting queue overflows.

The new standard accounts for this case. The switch can send the transmitting nodes (A or B) a special packet called PAUSE that tells these nodes to refrain from sending any new packets during a given period of time.

Switches relay incoming packets based on their knowledge of the network topology. They discover the port on which a node may be reached by sniffing the source address field of the incoming packets. This information suffices for all unicast packets. However, broadcast or multicast packets cannot be handled in the same way. By default, these packets are relayed on all ports except the port from which they were received. Since, in a given network, there may be more than a single path from one node to another, there is a risk that broadcast and multicast packets will start looping in the network. The same phenomenon may occur at network start-up, when the nodes have not yet learned the network topology and are thus obliged to relay the unicast packets in the same way they relay broadcast packets. To avoid loops, Ethernet switches implement an algorithm called "spanning tree" that deactivates all redundant links.

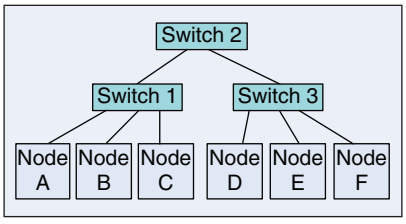

FIGURE 3 – Nodes interconnected using switches.

The Idea of Priority

The standard that describes the switch behavior caters to up to eight priority classes, even though most vendors limit their switches to two or four priority queues. The Ethernet packet holds a priority field with eight possible values (in an extension of 802.3, IEEE 802.1Q [41], a special service access point announces a priority header). When a packet is received on a switch, it is placed in the output queue of the output port that leads to the destination and that corresponds to the priority level of the packet. In each queue, packets are managed according to a first-come, first-serve policy. When a given output link is idle, the switch will select the oldest packet in the highest-priority output queue among those that are not empty. This is the packet that will be transmitted. If, immediately after the beginning of its transmission, a new packet with a higher priority is stored in a higher-priority output queue of the same output port, the new packet has to wait until the currently transmitted packet has been entirely transmitted. Thus, there is a potential blockage at each output port for the duration of the longest packet (often 1,518 B).

Toward a "Real-Time" Ethernet

Concurrently with the novelties introduced by the standards, a number of authors have proposed improvements to the temporal behavior of Ethernet (see [25] for more details) following three approaches: suppressing collisions, reducing the collision probability, and solving the collisions in a deterministic manner.

The first approach includes the use of switches or modifications to the MAC scheme. Collision probability may be lowered using reservations as in IEEE 802.11 (RTS-CTS mechanism), using the virtual CSMA approach [26]. CAN [13] and CSMA/DCR [27] are examples of deterministic collision resolution techniques. We shall not adhere to this classification because it does not show the degree of compatibility with "pure Ethernet" (the version as defined by the current IEEE standards).

Modifications to Ethernet may be categorized into two classes. The first category includes the proposals that alter the protocol in such a manner that a node that complies with the modified protocol cannot operate in the presence of unmodified nodes. The second class includes all the solutions that may work in a network that includes "pure Ethernet" nodes. Most of the solutions in the second class provide some form of temporal guarantees under the assumption that all nodes comply with the modified protocol. However, the guarantees vanish in the presence of unmodified nodes ("pure Ethernet" nodes). We will call the proposals in this subclass "homogeneous solutions." A second subclass includes all the solutions that keep the offered guarantees in the presence of "pure Ethernet" nodes. These will be called "heterogeneous solutions." These solutions are more interesting because they can

coexist with standard Ethernet devices (based on "pure Ethernet").

The taxonomy can be disputed. It has the advantage of being able to consider the degree of compatibility of the solution, which is a promise for long durability. We could have categorized the solutions based on the degree of guarantees. This would not have been useful, though, as most of the time guarantees are provided under the assumption that no transmission error may occur. In the presence of transmission errors, which are rare but unavoidable, only statistical guarantees may be provided.

Solutions Not Compatible with "Pure Ethernet"

Here, we look at the modifications to the IEEE 802.3 MAC that render the solution incompatible with the unmodified version. Most of them add a new MAC on top of the IEEE 802.3 MAC. All the traffic, whether with temporal constraints or not, goes through this additional layer (Figure 4) that controls access to the transmission medium. At the higher layers of the protocol stack, real-time traffic is either handled using TCP and IP or by specially designed protocols.

Quite naturally, the solutions use all the varieties of deterministic protocols, such as time division multiple access (TDMA) [28], window reservation [29], master-slave [30], tokens [31], or temporal priorities [32]. Other authors have suggested the use of the MAC protocols of fieldbuses, such Profibus or WorldFIP [4], on top of Ethernet.

Proposals that employ a pure TDMA seem appealing at first glance because everything is deterministic; each node has its emission slot. A more careful scrutiny shows that determinism is quickly lost in the presence of errors and that the solutions are highly inefficient. Some studies (which have been done for the TTP/C protocol ported on top of Ethernet at 1 Gb/s, which is representative of pure TDMA solutions) show that a maximum efficiency of about 4% [28] can be reached. In other words, a maximum of only 4% of the bandwidth is usable to carry useful traffic. The reasons for this are diverse. Since each packet suffers from a delay in each switch, sufficient guard time should be kept between time slots that belong to different nodes. Consider the example depicted in Figure 3. Assume node A emits in a given slot. Assume node F is the owner of the next slot. Between the end of the emission of node A and the beginning of the emission of node F, it will be necessary to wait the sum of the packet transit times in all three switches. Generally speaking, the guard time shall be greater or equal to the sum of the transit delays on all the switches of the network. The transit delay in a switch is usually comprised of the time necessary to receive (or transmit) the packet and a delay that is independent of the packet length. Hence, the guard time may be much longer than the time necessary to emit a packet. The delays in the switches also diminish the accuracy of the clock synchronization algorithms, thus requiring a further increase in the guard periods. Finally, to cater for errors, a number of slots are left free. They are used to retransmit the packets that have not been properly received. To account for the worst case, in which each packet is in error, the number of free slots should be equal to the number of used slots to be able to retransmit once every packet. This doubles the number of required slots.

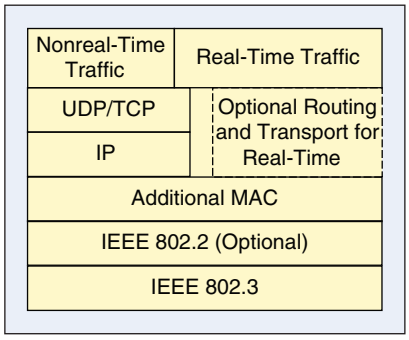

FIGURE 4 – Modified protocol stack with an additional layer.

Polling-based solutions (master-slave) can provide determinism. They are inefficient when the slave stations have highly different traffic requirements or when the traffic is irregular (sporadic). Most of the time, the master node will poll a slave station, and the slave station will answer that it has no traffic to send. A second source of inefficiency is the time spent between the reception of the request at a slave and the emission of the response. For instance, emitting a 64-B frame will take approximately 5 μs (at 100 Mb/s). The delay between the request and the response may easily be ten times larger; if the protocol is handled by software, it is difficult to have reaction times that are below a few tens of milliseconds. All this time is lost for communications.

Flexible time-triggered (FTT) Ethernet [33] combines TDMA and polling. As in WorldFIP, the entire communication is distributed over microcycles made of two parts: a synchronous window for guaranteed traffic and an asynchronous window for on-demand traffic. At the beginning of each microcycle, the master node broadcasts a packet that carries the identifiers of the data to be transmitted, at which point in time the node producing the data is allowed to transmit. Each node transmits the data in a packet at the designated instant without the need for a poll packet. All the consumers of the data can listen for the packet at that instant and capture the data. In the asynchronous window, the master node polls some slave stations to check if they have traffic to transmit. This solution exhibits the limitations of both polling and TDMA.

All the solutions that add a new MAC on top of the IEEE 802.3 MAC bring with them the limitations of that new MAC. Combined with Ethernet, they are inefficient. In addition, they cannot coexist with nodes that do not carry the same adjunction. These nodes do not comply with the additional access policy and will thus introduce errors causing the loss of the guarantees. The traffic generated by these nodes will also be declared illegal by the nodes conforming to the additional MAC. Therefore, it is necessary to isolate the modified nodes in a separate subnetwork connected with any other Ethernet network by a special bridge.

Modifications That Maintain Compatibility with "Pure Ethernet"

Homogeneous Solutions

Homogeneous solutions maintain the temporal guarantees only in the absence of other "pure Ethernet" nodes in the same subnet. Traffic smoothing and traffic shaping are good examples of such solutions. The idea behind traffic smoothing is that a flow that is regular in its arrival pattern has less chance to suffer from collisions. We also know that, for periodic traffic, it is allowed to occasionally loose a packet because the data the packet carries will be sent again at the next period and applications are tolerant to this kind of temporary loss. It is thus sufficient to offer probabilistic guarantees.

Traffic smoothing [34], [35] is based on the leaky bucket principle. The bucket has a maximum capacity of NMC credits and a refill period PR. Every PR unit of time, NMC credits are added to the bucket capacity. All that exceeds the maximum capacity is lost. A node needs at least one credit in its bucket in order to emit a packet. When a packet is emitted, a number of credits, equal to the size of the packet, are subtracted from the bucket capacity. According to authors who have studied traffic smoothing on top of Ethernet, as long as the total load on the network does not exceed some threshold, the packet loss ratio and the delay variations are drastically reduced compared with these performance measures for "pure Ethernet." Using the same principle on switched Ethernet, Jork Loeser reported the absence of losses and small delays in experiments with up to 98% load [36].

Heterogeneous Solutions

This subcategory includes all solutions that keep some temporal guarantees even in the presence of "pure Ethernet" nodes. EtheReal [37] is an example of such solutions. The key of the solution lies in the use of switches that are specially modified. Regular switches and hubs are prohibited. The end nodes use unmodified software stacks and hardware. Since, most of the time, real-time traffic is cyclic or periodic, for each real-time flow a connection is established between the sender and the first switch on the path. The switch reserves the resources necessary to guarantee the temporal constraints. The resource reservation is propagated to all the switches in the path leading from the source to the destination. The resource reservation protocol is the only addition on the nodes. A "pure Ethernet" node will not include this software; hence, it will not benefit from the possibility to reserve resources and obtain temporal guarantees for its traffic. All the traffic that has not been reserved is transmitted using a best-effort policy.

Using Ethernet combined with traffic smoothing and priorities as defined in IEEE 802.1D is a second solution. As explained earlier, switched Ethernet eliminates all the collisions. There is still a risk of overflow in the switch queues. Using traffic smoothing can drastically reduce and even avoid these overflows. Finally, priorities are used to limit the non-real-time traffic. Switches are able to label with the lowest priority all the incoming traffic that comes from a given port or does not carry any priority field. The guaranteed traffic will be given a high priority and smoothed. The other traffic or the traffic coming from "pure Ethernet" nodes will be given the lowest priority. It is thus possible to provide some temporal guarantees to the real-time traffic even in the presence of "pure Ethernet" nodes. To support synchronization of actions, the solution may be coupled with a distributed clock synchronization algorithm, such as IEEE 1588 [38].

Different "Industrial Ethernet" Proposals

Numerous vendors and interest groups offer an industrial communication solution under the name "industrial Ethernet." Most of the proposals so far have been standardized by the International Electrotechnical Commission (IEC) under the IEC 61158 series of standards, and here we analyze the temporal guarantees these proposals can provide. (A good general description of these protocols is given in [1].)

All the standardized "industrial Ethernet" proposals have been added to the IEC 61158 that covers some of the standardized fieldbuses. Table 1 provides a list of the standardized solutions. "Industrial Ethernet" solutions are highlighted.

A number of these proposals do not alter, or add, to the MAC. Therefore, they provide little if any additional guarantees. This is the case with Ethernet/IP, MODBUS TCP, MODBUS RTPS, PROFINET CBA, and PROFINET IO CC-A. MODBUS RTPS tries to improve efficiency by using a publish-subscribe interaction model at the application layer. Although there might be some improvement by avoiding blocking of the application while waiting for data, the proposal does not provide any more guarantees than "pure Ethernet" does.

Some solutions also try to improve responsiveness by removing TCP/UDP and IP and restricting real-time traffic to a single subnet. This may give lower response times under low traffic but does not provide any guarantees even statistical. PROFINET SRT (part of PROFINET CBA) is an example of this class of solutions. The bright side of all the solutions mentioned so far is their complete compatibility and coexistence with "pure Ethernet" nodes. The solutions are not interoperable, however, because the application layers are different.

Some of the proposals adopt the additional layer principle (Figure 3). This is the case with Ethernet Powerlink (EPL), Ethernet for Plant Automation (EPA), P-Net on IP, VNET/IP, and TCnet. In EPL, as in FTT-Ethernet, time is divided in constant length microcycles (Figure 5). A special node called the "managing node" starts the cycle by sending a start-of-cycle packet that is used for synchronization. It then polls the nodes (called control nodes, or CNs) one after another. These nodes may only emit packets as a response to the corresponding managing node poll (master-slave). A given node may be polled in every microcycle or less frequently (every N cycles) according to a predefined configuration. The time left to the slave to

TABLE 1—FIELDBUS AND "INDUSTRIAL ETHERNET" STANDARD UNDER IEC 61158.

COMMUNICATION PROFILE (CP) IN IEC 61784			IEC 61158 TYPES TO CORRESPONDING TO CP	
FAMILY CPF #	**TECHNOLOGY NAME**	**61784 PART #**	**CP NUMBER**	**TYPE #**
1	FOUNDATION FIELDBUS	1, 5–1	CP 1/1 FF H1	1,9
		1, 5–1	CP 1/2 FF HSE	5
		1, 5–1	CP 1/3 FF H2	1,9
2	CIP	1, 5–2	CP 2/1 ControlNet	2
		1, 2, 5–2	**CP 2/2 Ethernet/IP**	–
		1, 5–2	CP 2/3 DeviceNet	2
3	PROFIBUS & PROFINET	1, 5–3	CP 3/1 PROFIBUS DP	3
		1, 5–3	CP 3/2 PROFIBUS PA	3
		1, 5–3	**CP 3/3 PROFINET CBA**	10
		2, 5–3	**CP 3/4 PROFINET IO CC-A**	10
		2, 5–3	**CP 3/4 PROFINET IO CC-B**	10
		2, 5–3	**CP 3/4 PROFINET IO CC-C**	10
4	P-NET	1, 5–4	CP 4/1 P-NET RS-485	4
		1, 5–4	CP 4/1 P-NET RS-232	4
		2, 5–4	**CP 4/1 P-NET on IP**	4
5	WorldFIP	1, 5–5	CP 5/1 WorldFIP	7
		1, 5–5	CP 5/2 WorldFIP with subMMS	7
		1, 5–5	CP 5/3 WorldFIP minimal for TCP/IP	7
6	INTERBUS	1, 5–6	CP 6/1 INTERBUS	8
		1, 5–6	**CP 6/2 INTERBUS TCP/IP**	8
		1, 5–6	CP 6/3 INTERBUS minimal (subset of CP 6/1)	8
		2, 5–6	CP 6/4	8
		2, 5–6	CP 6/5	8
		2, 5–6	CP 6/6	8
8	CC-Link	1, 5–8	CP 8/1 CC-Link/V1	18
		1, 5–8	CP 8/1 CC-Link/V2	18
		1, 5–8	CP 8/1 CC-Link/LT	18
9	EPL	–	**CP 13/1**	–
10	VNET/IP	2, 5–10	**CP 10/1**	17
11	TCnet	2, 5–11	**CP 11/1 TCnet**	11
		2, 5–11	CP 11/2 TCnet-Loop	11
12	EtherCAT	2, 5–12	**CP 12/1**	12
		2, 5–12	**CP 12/2**	12
14	EPA	2, 5–14	**CP 14/1**	14
		2, 5–14	**CP 14/2**	14
15	MODBUS-RTPS	2, 5–12	**CP 15/1 MODBUS-TCP**	15
		2, 5–12	**CP 15/2 MODBUS-RTPS**	15
16	SERCOS	1, 5–16	CP 16/1 SERCOS I	16
		1, 5–16	CP 16/2 SERCOS II	16
		2, 5–16	**CP 16/3 SERCOS III**	19

respond is bounded. Its default value is 8 μs, though the exact value is device specific and is made available to the managing node. The packet sent by the slave as a response to a poll is transmitted in broadcast. All other nodes can then read it. A producer-consumer model may hence be easily built. Some space is left at the end of the microcycle. This is used by the managing node to poll nodes for non-real-time traffic. EPL bears a number of similarities with the WorldFIP fieldbus basic principles.

In the absence of transmission errors, EPL can provide a maximum bound to the transmission time of some piece of data from a producer to a consumer. This bound is equal to the polling period of the data plus the actual emission time of the packet that carries the data. Errors make the behavior highly unpredictable. In particular, any temporal error—a node emitting too late, for instance—will cause a collision. EPL provides some mechanisms to detect such errors, but temporal guarantees are lost. Additionally, as previously indicated, the polling principle is highly inefficient. The response time of the slave is one of the major factors in this inefficiency. The default response time corresponds to more than the duration of a minimal-length (64 B) packet. It will not be possible to exploit a network operating at 100 Mb/s at more than 25% of its capacity. (This percentage corresponds to a useful packet of 5 μs that requires a request of 5 μs and a gap of 10 μs (8 μs at the CN and 2 μs at the managing node). It should be noted that such low response times may only be achieved using hardware-based implementations that necessitate special silicon.) The performances will be even worse when switches are used (which is discouraged by EPL). Finally, non-real-time traffic suffers from very long latencies. Any node that requests non-real-time traffic has to declare it in its response to periodic pollings. The managing node will later poll the requesting node in the asynchronous part of the microcycle. Since only one node may be polled in a given microcycle, the waiting time may be quite long.

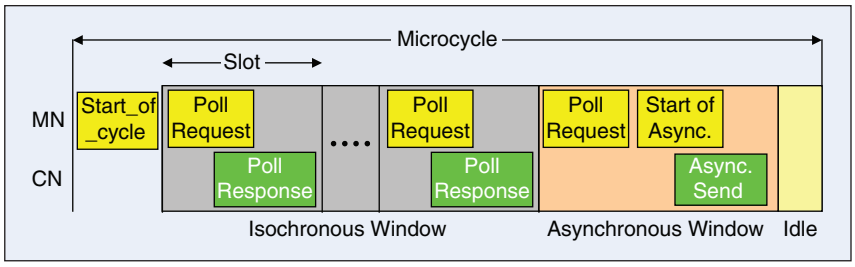

FIGURE 5 – Ethernet Powerlink microcycle.

EPA uses a pure TDMA scheme over fixed length cycles. The cycle is divided into two parts (Figure 6), one for the synchronous (periodic or cyclic) traffic and one for the sporadic (on-demand) traffic. In the synchronous part, each node is assigned by static configuration a point in time at which it is allowed to transmit. In the packet it transmits, a node may indicate its need for sporadic traffic. The second part of the cycle is reserved to this traffic. Each node that has indicated its intent to transmit in the asynchronous window may participate. Medium access is distributed and governed by priorities. Compared with EPL, EPA improves the efficiency by removing polling. Synchronization between nodes is no longer ensured by a synchronization packet at the beginning of the cycle but is based upon the IEEE 1588 protocol [38]. The synchronized clocks give the beginning of the cycle and are used to avoid collisions. Synchronous transfers are thus guaranteed in the absence of errors. In case there is a transmission error, there is no immediate retransmission. Rather, the information is obtained from the packet transmitted at the next cycle. Note that each node transmits its synchronous traffic at the same frequency. This reduces the efficiency or limits the applicability of EPA to configurations in which all nodes use the same reporting period. VNET/IP also uses the TDMA principle and is very similar to EPA.

As explained earlier, EPL, EPA, and VNET/IP cannot operate in the presence of "pure Ethernet" nodes. In the first two solutions, the proponents have defined a special bridge that allows interconnecting an EPA or an EPL subnet with an Ethernet network.

PROFINET IO addresses remote input/output (IO) for automation devices. Exchanges between the automation device and all the IO devices are cyclic. PROFINET IRT (PROFINET IO CC-C) adds isochronous transfer with very low jitter. Each cycle is then divided into three windows, one for the isochronous traffic, one for alarms, and one for non-real-time traffic. The total traffic may not exceed 60% of

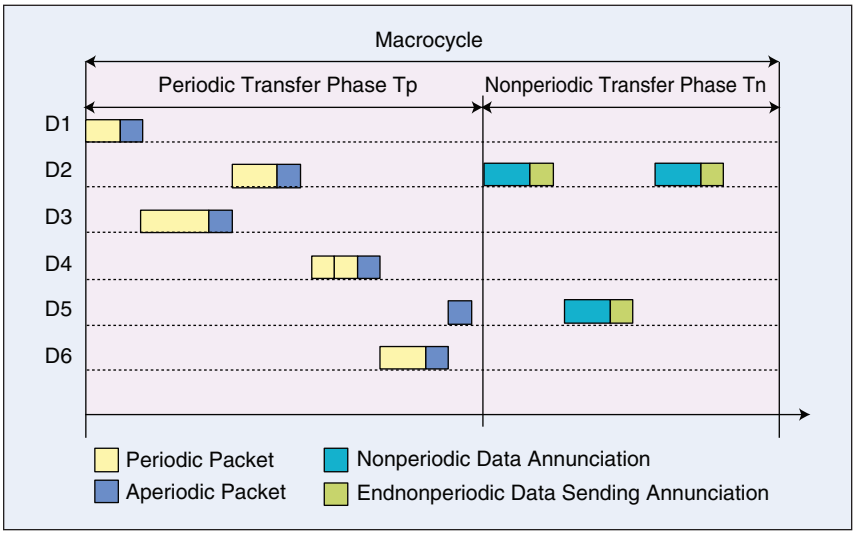

FIGURE 6 – EPA cycle organization.

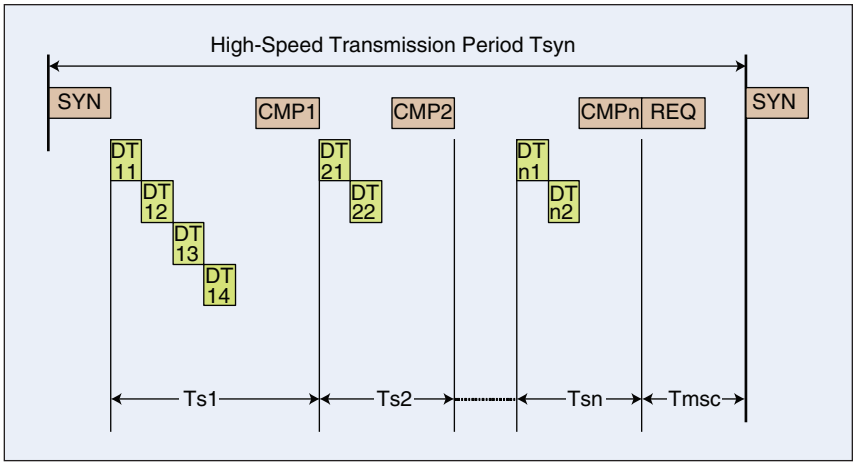

FIGURE 7 – TCnet cycle.

the cycle on the average. Each node is connected to a specially designed switch. The switch blocks all other traffic when isochronous traffic takes place. The switch also implements a cut-through policy and traffic priority handling. These techniques yield a very low jitter in the isochronous traffic and greatly improve the latency of high-priority traffic. The special switch is the key to real-time guarantees. All nodes in a network that must comply with real-time guarantees must be connected through switches of this kind. Due to traffic blocking and classification, it is possible to connect "pure Ethernet" nodes on the switches without altering the temporal guarantees of the isochronous traffic.

P-NET on IP and TCnet add a token passing mechanism on top of IEEE 802.3. P-NET on IP uses a virtual token based on silence on the medium. If the medium is free for more than a given delay, the token is implicitly passed to next node. To implement this, every station owns a counter that increments each time the medium is idle. When the counter value is equal to the node ID, the node can transmit one packet at most, including the response from the destination and possible retries. The destination node has a maximum bound in the time to respond, which is slightly less than the duration of the silence that induces the counter increase. As a result, P-NET on IP cannot use switches unless the response time is set higher than the sum of the transit delays on all switches (minimum 5 μs per switch at 100 Mb/s). This means that for each transfer of a data packet (5 μs, approximately), there is a silence that is much longer than the packet transfer. It is thus difficult to reach good efficiency with this solution. On the other hand, P-NET on IP has the advantage of being very simple and robust to losses. It may not coexist with "pure Ethernet" nodes, however.

TCnet uses an explicit token passing mechanism that passes in ascending order of node ID. All exchanges are organized in cycles that repeat (Figure 7). The cycle ends after all nodes have received the token and new nodes have been given the opportunity to join the network. Different classes of traffic can coexist: high-, medium-, and low-speed cyclic, as well as sporadic Ethernet, which corresponds to conventional TCP/UDP/IP traffic. Each time a node gets the token, the node is allowed to send high-speed traffic until some token hold time has elapsed. As long as the real token rotation time is lower than some target values (different for the different classes), the node may send the corresponding traffic. This node then explicitly passes the token to the next node. According to the standard, the cycle may be set between 0.1 and 160 ms. TCnet is certainly more general than P-NET on IP because it can handle traffic in a differentiated way and does not implicitly assume that all nodes carry the same quantity of traffic. The drawbacks of TCnet are a higher complexity and the well-known problems of the timed-token types of protocols, such as complexity or long inaccessibility when the token is lost [39]. Both proposals cannot operate in the presence of "pure Ethernet" nodes.

Two proposals, EtherCAT and SERCOS III, offer a quite different approach to the problem. They bear a number of similarities. Both start from the fact that the minimum size of an Ethernet packet (512 b) makes it impossible to send small quantities of information efficiently. In industrial automation, most of the time, real-time data are small, so it is quite interesting to overcome this inefficiency. Both proposals suggest concatenating data coming from different nodes in the same Ethernet packet. This principle is often used in the master-slave system to convey data from the master to the slaves but seldom vice versa. There are, however, older solutions, such as the Cambridge ring [42] and Interbus [5], that implement this. This principle is possible only in ring topologies, however. On the fly, nodes must modify the data they receive to update them with their new data. This principle also implies that the frame sequence check (CRC) must be updated on the fly. This is clearly impossible with standard NICs. In both proposals, all nodes have two Ethernet interfaces.

In EtherCAT, a packet is first transmitted by the master node. The packet bits are relayed from node to node (Figure 8). The last node closes the

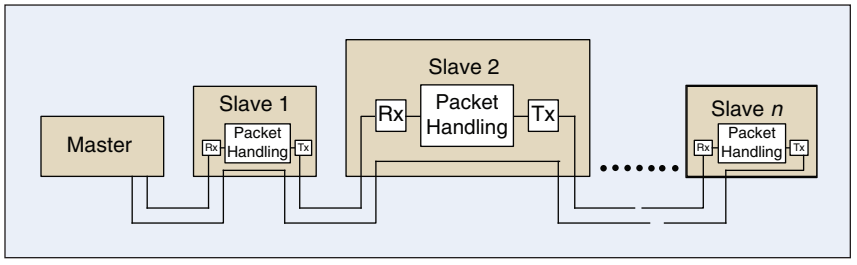

FIGURE 8 – EtherCAT topology and NIC architecture.

path, and the packet is sent back to the master. In the forward path (from the master to the nodes), the nodes have the capability to modify the packet. In the reverse path, the packet is not altered. In each packet, there are one or more fields that are assigned to each node. The node knows the position of the fields by static configuration. Some fields are reserved to the data from the master to the slave. Some carry the information from the slave to the master; these fields are left blank when the master emits the packet and are updated on the fly by the node to which the fields belong. The node also recalculates the CRC according to the updated fields. Contrary to SERCOS III, EtherCAT is not based on periodic exchanges of packets. Synchronized clocks are used to synchronize operations when needed. High synchronization accuracy is obtained because forwarding delays in the nodes can be calculated very precisely.

Any EtherCAT node can be directly connected to a "pure Ethernet" network. An EtherCAT node will be able to send and receive non-real-time traffic. Real-time traffic is possible only in EtherCAT segments, as depicted in Figure 9. In this topology, the entire EtherCAT segment behaves like a single Ethernet device that receives packets and responds with other packets. The packet carries the address on the first node in the EtherCAT as the destination address.

In EtherCAT, direct traffic from slave to slave is not possible; the master must relay it. SERCOS III removes this limitation. It allows nodes to inspect packets in the reverse path, as depicted in Figure 10. Using both Ethernet interfaces, it is also possible to build a double ring network, as in Figure 11. This redundancy may be used to improve dependability.

Since all packets may be read and updated in the forward as well as the reverse path, each node is able to read what another node has updated. In EtherCAT as shown in Figure 8, for example, if slave 2 updates some values in the packet, slave 1 will not be able to read the modified values. With SERCOS III, this becomes possible because

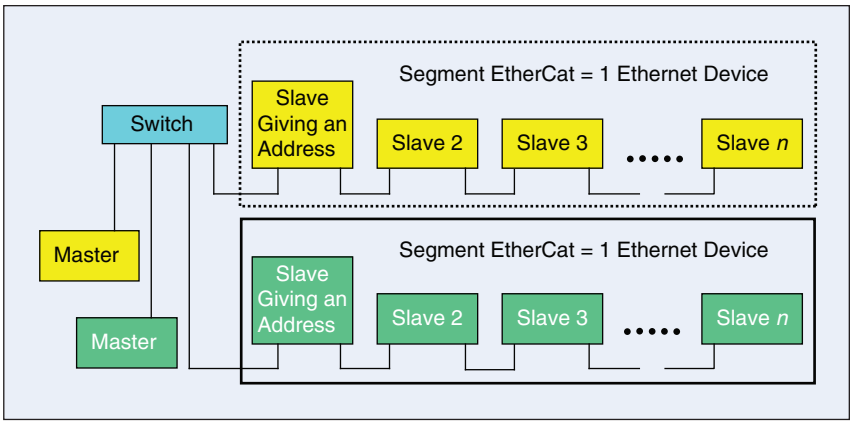

FIGURE 9 – EtherCAT topology when connected to a "pure Ethernet" network.

the packet can be read when returning back to the master node. Slave-to-slave communication becomes possible and is provided by SERCOS III.

In addition, the standard provides the possibility to send IP packets from one slave to another using the following principle. SERCOS III operates periodically with a constant duration cycle (minimum 31.25 μs) and very low jitter (1 or 50 μs depending on the selected class). The cycle is divided into windows. Some windows are dedicated to synchronous traffic carried by packets sent by the master and passed from slave to slave as in EtherCAT. One of the windows is open to IP packets. During this asynchronous window, slaves that have IP packets to send open the ring. They send the pending packets to both of their Ethernet ports at the same time. During the same window, any incoming packet at a slave is stored in the node. If its destination is the node, the packet is passed to the higher layers. If its destination is another node, the packet will be retransmitted to the other port as soon as the port is free, provided the asynchronous window has not ended. Slave-to-slave communication of IP packets is hence possible. It is also possible to connect a SERCOS III slave node directly to a "pure Ethernet" network. One of the node's ports must then be left free.

EtherCAT and SERCOS III support high-performance applications. Cycle times can be very short. EtherCAT is able to send and receive traffic from 100 slave nodes in less than 0.1 ms when there are no errors. For real-time

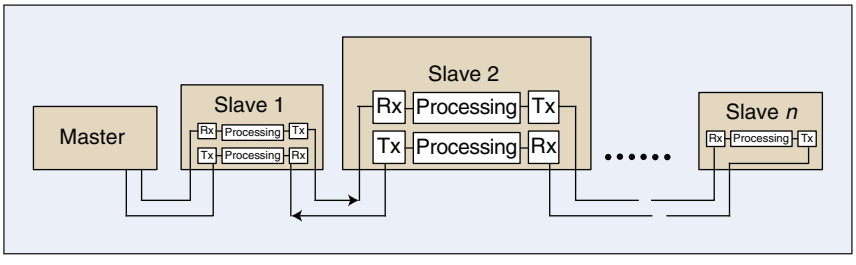

FIGURE 10 – SERCOS III topology and NIC architecture.

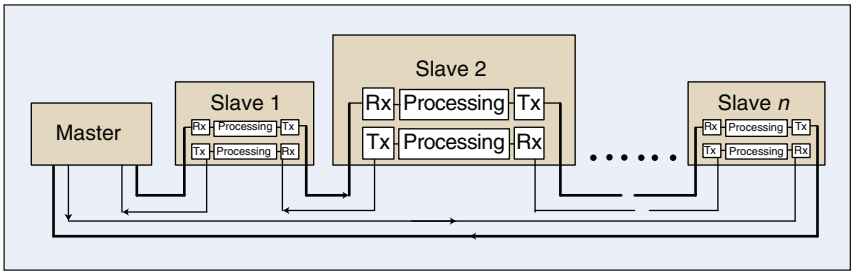

FIGURE 11 – SERCOS III double ring topology (primary ring in thick lines, secondary ring in light lines).

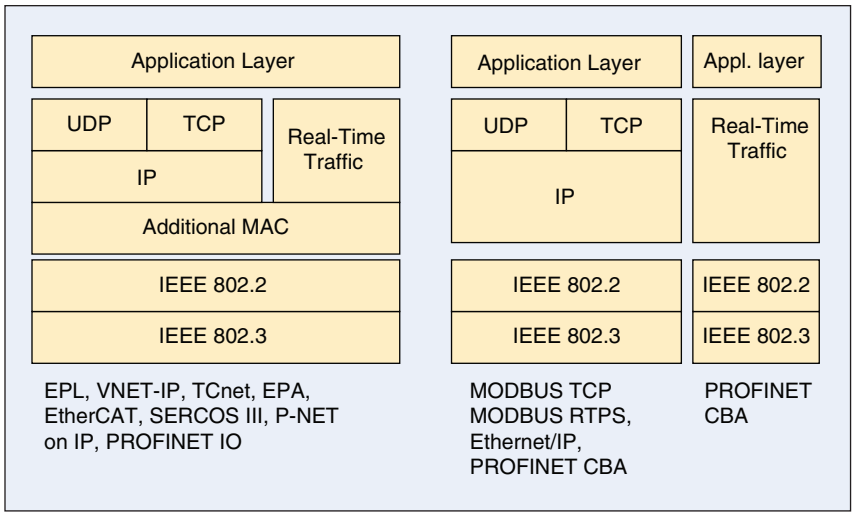

FIGURE 12 – Classification of the standards according to their architectures.

traffic, both SERCOS III and EtherCAT do not include any immediate error correction by retransmission. Rather, errors are corrected with the new values sent at the next cycle. SERCOS III provides periodicity and controlled jitter. This is not the case with EtherCAT, which leaves this to the designer of the master node. Synchronized actions are still possible, however, using synchronized clocks. Nodes complying with one of these solutions may be connected to "pure Ethernet" networks, but they loose all the temporal guarantees in such a case.

Figure 12 provides an overview of the stack architectures of the proposals. The proposals that do not add another MAC on top of IEEE 802.3 are obviously compatible with "pure Ethernet." They are also the solutions that provide only "best effort" guarantees. Some of them adopt the publish-subscribe model with the advantage of temporally decoupling the communicating nodes. There is thus no time lost in waiting for the answer of the server.

The solutions that offer more or less strict temporal guarantees all rely on a modified stack with additional medium access rules. SERCOS III provides the highest performances. As indicated in Table 2, all the solutions but PROFINET IRT no longer offer guarantees when mixed with "pure Ethernet" nodes. PROFINET IRT relies on a special switch to isolate isochronous traffic from non-real-time traffic. The switch is also designed to limit jitter to a minimum.

A Solution Based Entirely on Existing Standards

All the proposed solutions that are able to provide temporal guarantees more or less depart from the IEEE 802.3 standard. Going back to the ideas presented earlier regarding homogeneous solutions, we will now sketch how a real-time solution may be designed around existing IEEE standards.

TABLE 2—CLASSIFICATION OF THE SOLUTIONS ACCORDING TO THEIR COEXISTENCE CAPABILITY.

CLASS	BEHAVIOR IN PRESENCE OF 802.3 COMPLIANT NODES	STANDARDS
Noninteroperable	Do not operate properly	SERCOS III, TCnet, EPA, EPL, P-NET on IP, VNET-IP, EtherCAT
Interoperable homogeneous	Operate properly but loose temporal guarantees	EtherCAT**, SERCOS III **
Interoperable heterogeneous	Operate properly and keep temporal guarantees	PROFINET IRT, Ethernet IP*, MODBUS TCP*, MODBUS RTPS*, PROFINET CBA*

* These protocols do not offer any special guarantee even in the absence of other nodes.
** Individual nodes in Ethernet mode.

As shown by Loeser and Härtig [36], switched Ethernet combined with traffic smoothing is able to avoid any buffer overflow in the switches even for network loads close to 100%. With this approach, it is possible to provide guarantees in terms of throughput, latency, and delay variation. However, the presence of nodes that do not implement traffic smoothing destroys these guarantees. Coupling the approach with the use of IEEE 802.1D and IEEE 802.1Q priorities is a solution. All traffic coming from nodes that do not implement traffic smoothing is given the lowest priority and thus has limited impact on the real-time traffic. Traffic smoothing combined with priorities is a good solution for many applications.

The only problem with this approach is that it is not possible to accurately predict when a packet will be transmitted. Sending packets at precise instants is used by some proposals, such as EPL and SERCOS III, to synchronize actions on different nodes. Some papers even propose that periodic transmission and low jitter are mandatory to synchronize actions. In fact, this can be achieved in a different way. Actions can be synchronized using distributed synchronized clocks as in EtherCAT. Instead of triggering the actions when a given packet is received, the actions are performed at a precise instant given by the clock on each node. Because the clocks are synchronized—using IEEE 1588 [38], for instance—actions may be precisely synchronized without the need for periodicity and low jitter. In case these actions require data from other nodes, the only constraint is that the data should be transmitted prior to the action.

Conclusion

The name "industrial Ethernet" covers different proposals to use Ethernet (IEEE 802.3) as a support for communications on the factory floor. We have analyzed all the proposals that were recently standardized by the IEC under IEC 61158. A number of solutions are just as fast as "fast Ethernet" and do not provide any

temporal guarantees. Some of them are quite high-performing and provide some level of guarantees. A few exhibit very high performances and guarantees. They all provide an application layer dedicated to industrial communications.

However, very few solutions offer guarantees when different kinds of errors, transmission, timing, losses, and other such things, are taken into account. As a matter of fact, it becomes much more difficult to calculate temporal guarantees in the presence of errors. It has to be noted that, with the exception of one, all solutions that provide guarantees are not compatible with Ethernet and require software and even hardware modifications. PROFINET IRT is able to provide temporal guarantees in the presence of "pure Ethernet" nodes. It relies on a special switch to offer strict guarantees, particularly of jitter.

In the early years of the fieldbus domain, a few initiatives tried to come up with a single solution. This was certainly a dream. We now have a completely opposite trend with more than 23 proposals using Ethernet. Even for already standardized proposals, improvements are still coming (see, for instance, [43]). This means that we have not yet reached maturity. Let us hope that reason will prevail and a few solutions will emerge.

In the absence of very strict jitter constraints and extremely low latencies, the solution may reside in an adequate combination of existing IEEE standards, which would avoid defining new and incompatible standards. This would keep the solution completely compatible without the need for stack modifications. An adequate application layer, however, would be needed.

Biography

Jean-Dominique Decotignie is with the Centre Suisse d'Electronique et de Microtechnique SA in Neuchatel, Switzerland. He is a Fellow of the IEEE.

References

[1] M. Felser, "Real-time Ethernet—Industry prospective," *Proc. IEEE*, vol. 93, no. 6, pp. 1118–1129, June 2005.

[2] J. Kurose, M. Schwartz, and M. Yemini, "Multiple-access protocols and time-constrained communication," *ACM Comput. Surv.*, vol. 16, no. 1, pp. 43–70, Mar. 1984.

[3] J. Boudenant, B. Feydel, A. Peyrache, and P. Rolin, "Lynx: An advanced deterministic CSMA-CD local area network prototype," in *Proc. Advanced Seminar on Real-Time Local Area Networks*, Bandol, France, Apr. 16–18, 1986, pp. 177–192.

[4] *Digital Data Communications for Measurement and Control—Fieldbus Standard for Use in Industrial Control Systems. Parts 1 to 6*, IEC Standard 61158, 2003.

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

[6] *Low-Voltage Switchgear and Controlgear—Controller-Device Interfaces (CDIs)—Part 2: Actuator Sensor Interface (AS-i)*, IEC Standard 62026-2, 2000.

[7] *Electrical Equipment of Industrial Machines—Serial Data Link for Real-Time Communication Between Controls and Drives*, IEC Standard 61491, 2002.

[8] *Control Network Specification*, Electronic Industries Alliance Standard EIA-709.1, Mar. 1998.

[9] H. Kirrmann and P. Zuber, "The IEC/EEE train communication network," *IEEE Micro.*, vol. 21, no. 2, pp. 81–92, Mar./Apr. 2001 (see also IEEE Std. 1473-1999).

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

[11] *Low-Voltage Switchgear and Controlgear—Controller-Device Interfaces (CDIs)—Part 3: DeviceNet*, IEC Standard 62026-3, 2000.

[12] *Low-Voltage Switchgear and Controlgear—Controller-Device Interfaces (CDIs)—Part 5: Smart Distributed System (SDS)*, IEC Standard 62026-5, 2000.

[13] *Road Vehicles—Exchange of Digital Information—Controller Area Network (CAN) for High-Speed Communication*, ISO Standard 11898, 1993.

[14] *Information Processing Systems—Open Systems Interconnection—Basic Reference Model: The Basic model*, ISO/IEC Standard 7498-1, 1994.

[15] J.-P. Thomesse, "Time and industrial local area networks," in *Proc. 7th Annu. European Computer Conf. on Computer Design, Manufacturing and Production (COMPEURO'93)*, Paris-Evry, France, May 24–27, 1993, pp. 365–374.

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

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

[18] 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.

[19] L. Svobodova, "Implementing OSI systems," *IEEE J. Select. Areas Commun.*, vol. 7, no. 7, pp. 1115–1130, Sept. 1989.

[20] ISO/IEC (1997). 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 Technical Report 11802-5: 1997(E).

[21] *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 ed. (revision to IEEE Std 802.3, 2000 ed).

[22] D. Boggs, J. Mogul, and C. Kent, "Measured capacity of an Ethernet: Myths and reality," in *Proc. SIGCOMM'88*, 1988, pp. 222–234.

[23] K. Ramakrishnan and H. Yang, "The Ethernet capture effect: Analysis and solution," in *Proc. 19th Conf. on Local Computer Networks*, Minneapolis, MN, Oct. 2–5, 1994, pp. 228–240.

[24] *IEEE Standards for Local and Metropolitan Area Networks: Media Access Control (MAC) Bridges*, IEEE Standard 802.1D, 1990.

[25] J.-D. Decotignie, "Ethernet-based real-time and industrial communications," *Proc. IEEE*, vol. 93, no. 6, pp. 1102–1117, June 2005.

[26] 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, Sept. 1985.

[27] G. Le Lann, *A deterministic Multiple CSMA-CD Protocol*, Internal Report of INRIA Project, PRO-1-002, Jan. 1983.

[28] M. Schwarz, "Implementation of a TTP/C cluster based on commercial gigabit Ethernet components," Diplomarbeit, Technischen Universität Wien, Dec. 2002.

[29] 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, July 1980.

[30] Z. Wang, G. Xiong, J. Luo, M. Lai, and W. Zhou, "A hard real-time communication control protocol based on the Ethernet," in *Proc. 7th Australasian Conf. on Parallel and Real-Time Systems*, Sydney, Australia, Nov. 29–30, 2000.

[31] 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. on Consumer Electronics*, Los Angeles, CA, June 2–4, 1998, pp. 28–29.

[32] D. W. Pritty, J. R. Malone, D. N. Smeed, S. K. Banerjee, and N. L. Lawrie, "A real-time upgrade for Ethernet based factory networking," in *Proc. Int. Conf. on Industrial Electronics, IECON*, vol. 2. Orlando, FL, Nov. 6–10, 1995, pp. 1631–1637.

[33] P. Pedreiras, P. Gai, L. Almeida, and G. C. Buttazzo, "FTT-Ethernet: A flexible real-time communication protocol that supports dynamic QoS management on Ethernet-based systems," *IEEE Trans. Ind. Inform.*, vol. 1, no. 3, pp. 162–172.

[34] S. Kweon and K. Shin, "Statistical real-time communication over Ethernet," *IEEE Trans. Parallel Distrib. Syst.*, vol. 14, no. 3, pp. 322–335, Mar. 2003.

[35] L. Lo Bello, G. A. Kaczynski, O. Mirabella, "Improving the real-time behavior of Ethernet networks using traffic smoothing," *IEEE Trans. Ind. Inform.*, vol. 1, no. 3, pp. 151–161, Aug. 2005.

[36] J. Loeser and H. Härtig, "Low latency hard real-time communication over switched Ethernet," in *Proc. 16th Euromicro. Conf. on Real-Time Systems*, Catania, Italy, June 30–July 2, 2004, pp. 13–22.

[37] S. Varadarajan and T. Chueh, "EtheReal: A host-transparent real-time fast Ethernet switch," in *Proc. 6th Int. Conf. on Network Protocols*, Austin, TX, Oct. 13–16, 1998, pp. 12–21.

[38] *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Standard 1588, 2002.

[39] B. Upender and P. Koopman, "Communication protocols for embedded systems," *Embedded Syst. Program.*, vol. 7, no. 1, pp. 46–58, Nov. 1994.

[40] J.-P. Thomesse, "Fieldbus technology in industrial automation," *Proc. IEEE*, vol. 93, no. 6, pp. 1073–1101, June 2005.

[41] *IEEE Standards for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks*, IEEE Std. 802.1Q-1998.

[42] M. Wilkes and D. Wheeler, "The Cambridge digital communications ring," in *Proc. Local-Area Commun. Netw. Symp.*, Boston, USA, May 1979.

[43] M. Schumacher, J. Jasperneite, and K. Weber, "A new approach for increasing the performance of the industrial Ethernet system PROFINET," in *Proc. 7th IEEE Int. Workshop on Factory Communication Systems (WFCS 2008)*, Dresden, Germany, May 2008, pp. 159–167.