Real-Time Systems

Lucia Lo Bello[1], José Alberto Fonseca[2], Wilfried Elmenreich[3]

[1]Department of Computer and Telecommunications Engineering, University of Catania, Italy
[2]Department of Electronics, University of Aveiro, Portugal
[3]Institute of Networked and Embedded Systems, Klagenfurt University, Austria

1 Introduction on Real-time systems

1.1 Definition

A real-time system is a system in which the correctness of the results does not only depend on the logical correctness of the computations/operations performed to obtain them, but also on the time the results are obtained [Kop97,Sta88]. Logically correct, but late results can be either not useful or even harmful to the system, depending on the nature of the application considered. Examples of real-time applications are found in industrial plants control, automotive, avionics, industrial automation, robotics, monitoring systems, multimedia systems, telecommunications, interactive systems, consumer electronics, etc. Unlike conventional, non-real-time computer systems, real-time computer systems are closely coupled with the physical process or the environment being monitored and controlled.

A *real-time system* (Figure 1) consists of a real-time computer system, a controlled object and may also feature an operator. The real-time computer system must react to stimuli from the controlled object (or the operator) within a time interval specified by the *deadline*. The concept of deadlines must not be confused with fast or high performance computing. Real-time

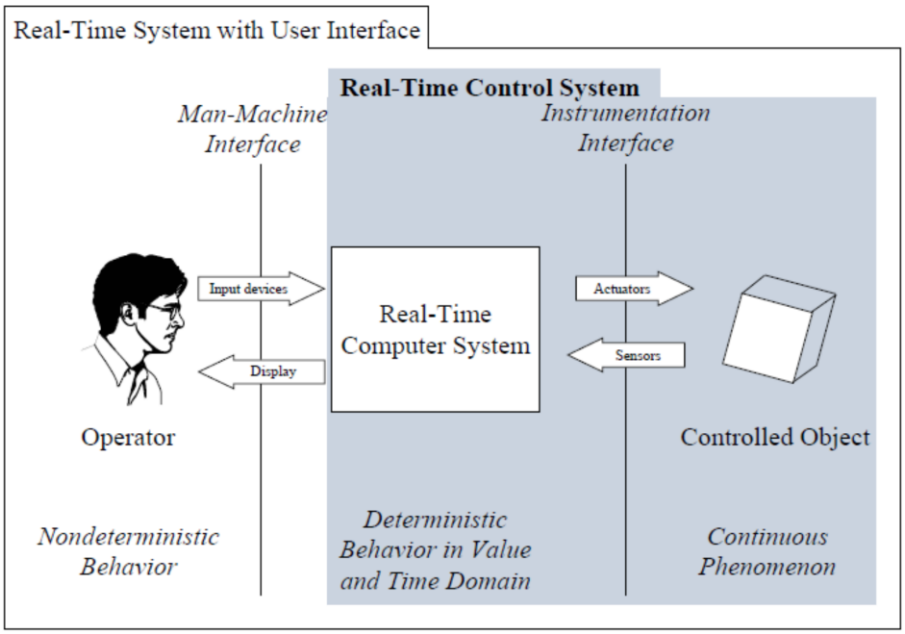

Figure 1 Elements of a Real-Time System

computing is different since the objective of fast computing is to minimize the *average* response time of a task, while real-time computing is concerned about the *maximum* or *worst-case* response time and the difference between minimum and maximum response time, the so-called *jitter* [Sta88]. The most important property of a real-time system is not high speed, but *predictability* [Sta90], which means the ability to determine in advance whether the task can be completed within its deadlines or not.

1.2 Real-time constraints characterization

A criterion commonly used to classify the timing constraints of real-time tasks is the usefulness of their results (value) as functions of the tasks' completion time and, in particular, of their lateness. The lateness of a task is defined as the difference between the task actual completion time and its deadline. If the task is on time, its lateness will be zero or negative. If the task is late, a positive lateness will be found. To what extent a deadline miss will compromise the system depends on the real-time nature of the deadlines imposed on the system. When missing a deadline can have catastrophic consequences to the system, the deadline is called hard. If a deadline miss only entails a performance degradation, but does not jeopardize the correct system behaviour, the deadline is called soft.

According to this criterion, three types of tasks can be identified, i.e., *hard*, *soft* and *firm* real-time. All these kinds of tasks give a positive *value* to the system if they complete on time. The difference between the three categories becomes significant when the tasks are late. If the late task is *hard* real-time, the value of the result produced by the task after its deadline is negative, thus indicating that a damage on the system occurs. As a result, hard real-time tasks have to meet their deadlines under any circumstances. Conversely, the value of the result produced by a *soft* real-time task starts decreasing as the lateness of the task becomes positive and will eventually become null at a given point of time after the deadline. So, the late completion of a soft real-time task can be occasionally tolerated, and it is advisable to minimize the lateness of the task, as the value of a late task decreases with time. Finally, the value of the result produced by a *firm* real-time task drops abruptly to zero as soon as the lateness becomes zero and remains null with increasing lateness values. As it brings no value to the system, a firm real-time task should be either completed on time or dropped if it is late, as there is no point to continue its execution after the deadline.

In the case of hard deadlines, suitable validation techniques are required to demonstrate that the system adheres to the intended timing behaviour. If the deadlines are soft, in general no validation is required and it is sufficient to prove that it is possible to meet a timing constraint specified in terms of statistical averages (e.g., that the average deadline miss rate is below a given application-dependent threshold).

1.3 Typical application domains

Real-time systems can be found in many different application domains. However, unlike personal computers (PCs) and workstations where users are fully aware of (non-real-time) applications such as email, Internet browser and text processing, real-time applications are often implemented as *embedded systems* providing their service hidden from our view [Elm09]. The typical application domains for real-time systems are:

- Digital control systems implementing a feedback loop from one or more sensor readings to one or more actuators. Thus, the controlled parameter is kept at a desired value even when conditions change in the system. Since such control loops are typically very sensitive to jitter and delay, a digital control system has to be implemented in a way that the delay between measurement and actuating is small in comparison to the timing parameters of the controlled system. Jitter is especially critical in such applications. Digital control systems are hard real-

time systems. Violation of the timing requirements can result in instabilities of the control loop and, therefore, often in damage of physical systems. An example is the ignition control system in a car's combustion engine. The requirement of this application is basically to make the fuel-air mixture ignite in the cylinder so that the piston is accelerated on its way down. Now imagine a badly designed control system that may eventually delay the ignition instant by a few milliseconds – the fuel-air mixture is now ignited while the piston is in the wrong position. This small time delay changes the forces to piston, connection rod and crankshaft in a way that may permanently damage the engine.

- Man-machine based real-time systems. The real-time control system incorporates the instrumentation interface consisting of input devices and output devices that interface to a human operator. Across this interface, the real-time computer system is connected to the process environment via sensors and actuators that transform the physical signals of the controlled object into a processable form and vice versa.
- High level controls involve flight management systems, factory automation, robot control, etc. The real-time tasks require, for example, path and trajectory planning. The real-time control system is required to provide its results within a given maximum response time. Many of these applications, e.g., in transportation or plant automation, are critical requiring a highly dependable implementation of the real-time control system.
- Many signal processing applications come with real-time requirements, needing to provide throughput at a given sampling rate. Voice processing applications for telephony further require that the processing delay does not exceed a given maximum. This kind of application often has soft real-time requirements, since a violation of the timing merely diminishes the quality of the service. Other soft real-time applications include video conference applications, online gaming, and, to some extent, instant messaging.

1.4 Real-time Scheduling and relevant metrics

A *schedule* is a mapping of task executions on a resource. For example, if the tasks are processes, the resource will be the processor, while if the tasks to be performed are data packet transmissions, the resource will be the network bandwidth. A schedule is *feasible* if all the tasks complete their execution under a set of specified constraints. A task set is *schedulable* if there exists a feasible schedule for it. In the following, like in [Liu00], we define a *task* as a sequence of *jobs* which jointly provide a system function, while a *job* is the unit of work that is scheduled and executed by the system. This notation allows us to deal with different system activities, such as, the execution of a process on a processor or the transmission of a data packet on a network, in a uniform way.

While in conventional non-real-time systems the commonly-used metrics are throughput, fairness, and average response times, in real-time scheduling all the target metrics are relevant to the system timeliness. Typical metrics are therefore:

- Response time, defined as the difference between the completion time and the release time of a job;
- Absolute jitter, defined as the difference between the minimum and the maximum response time for a job;
- Maximum lateness for soft real-time tasks, defined as the maximum difference between the finishing time of a job and its deadline;
- Deadline miss ratio for soft real-time tasks, i.e., the ratio between the number of jobs which missed their deadlines and the overall number of soft real-time jobs released;
- Throughput on time, defined as the amount of jobs completed by the deadline per time unit;
- Cumulative value for a set of soft or firm real-time tasks, defined as the sum of all the values calculated at each job completion times, given their value functions.

Since the seminal paper of Liu and Layland [Liu73], a very large amount of works dealt with real-time scheduling, addressing it in multiple scenarios and from different perspectives. Interested readers may refer to [Liu00] and [Sha04].

2 Real-time communication

2.1 Deterministic vs. statistical communication

In hard real-time systems, deadline miss should never happen, as deadline misses severely affect the application behaviour and determine non-recoverable errors. In soft real-time systems, occasional violations of deadlines can be tolerated, although at the expense of a degradation of the *Quality of Service* (QoS) provided to the application. Offering real-time support on networks means that a predictable time behaviour of communications can be guaranteed, either in deterministic or in stochastic way. Predictable time behaviour is not necessarily a synonym of constant time behaviour. What really counts is the existence of an upper bound on some relevant QoS index, such as, end-to-end packet delivery time, packet channel access time, roundtrip time, etc. The existence of such an upper bound makes it possible to assess whether the packet deadlines will be met or not, in either deterministic or stochastic way. It is therefore possible to define a

- Deterministic real-time channel as one that provides a priori deterministic guarantees for the timely delivery of packets between two end-points.
- Statistical real-time channel as one that guarantees the timely delivery of packets between two end-points in statistical terms, i.e. that the probability that a packet misses its deadline is less than a certain loss tolerance Z, i.e. $Pr(\text{packet delay} > \text{delay bound}) \leq Z$.

Statistical guarantees on deadline meeting satisfy the requirements of real-time non mission-critical applications, such as periodic control or industrial multimedia, which can tolerate a small violation probability with delay bounds. In an automated manufacturing system, for example, real-time periodic control messages may need to be delivered within 20 ms of their generation with a 98% probability, while voice packets may require delivery within 40 ms with a 94% probability, and so on. Deterministic and statistical analysis techniques for wired industrial networks have been thoroughly investigated in the literature. For instance, in [Kwe03] Shin et al. analytically demonstrate that, in order to statistically bound the medium access time for a real-time packet transmitted over Ethernet networks in the presence of both real-time (RT) and non-real-time (NRT) traffic, it is sufficient to keep the total arrival rate for new packets generated by stations below a threshold called the *network-wide input limit*. To enforce such a limit on a per-station basis, each station is assigned a local threshold, called a *station input limit*, and a middleware called a *traffic smoother* is entrusted with the regulation of the outgoing NRT stream on each node. Several traffic smoothers have been proposed in the literature, for either Shared or Switched Ethernet, which differ in the way the station input limit is enforced. More details can be found in [Kwe04] and [LoB05]. Among recent approaches to obtain RT performance over an Ethernet there are the FTT_Ethernet [Ped05], the PEAC protocol [Bon03], the Time-Triggered Ethernet [Kop05].

2.2 Best effort vs. guaranteed service

Guaranteed service means that the user is given a *guarantee* on the system timing behaviour (for instance, on a message end-to-end delay and/or jitter, medium access time, etc.) and that the meeting of the relevant timing constraints has to be validated, through formal methods or exhaustive simulation and testing. This service is required for hard real-time traffic. On the contrary, *best-effort service* means that, although the user requires the best quality of service the system can provide, the system is allowed to deliver a lower quality than expected. This service is suitable for soft real-time traffic.

2.3 Performance metrics

In real-time communication the primary performance metrics are related to the timeliness of data exchange over the network. As real-time flows have to be provided with a different QoS than non-real-time ones, when different types of traffic have to be transmitted over the same channel, traffic prioritization is required. In particular, end-to-end real-time performance of real-time traffic is very important, as usually networked real-time systems handle monitoring and control applications, which typically require a response within *bounded delay*, often combined with *low jitter* values.

Typically, the end-to-end delay is composed of several stages of data processing and transmission. To be able to provide a bounded delay, the delay of each stage must be bounded. For example, the queuing delay in the network, which in turn depends on the queuing delay in the local queues and on the number of hops in the path from the source to the destination node of the packet in packet switched networks. In each network device traversed by the packet, multiple queues with different priorities are needed to handle different traffic classes and real-time scheduling algorithms, such as Weighted Fair Queuing [Dem90], or non-preemptive versions of Rate Monotonic (RM) or Earliest Deadline First (EDF) [Liu73] can be exploited to deal with real-time traffic and calculate delay bounds for it. However, WFQ introduces a significant computational overhead, while the effectiveness of priority queuing depends on the number of priorities, which can be defined (the finer the granularity, the better the resulting traffic differentiation). Even the capacity of the queues of the network devices the packet traverses on its path and the total network load have to be taken into account, as packet dropping may occur if any of these queues fills up. Suitable flow control algorithms, selective discarding, and packet marking policies are needed to support QoS for real-time traffic. Another significant contribution to the end-to-end delay is given by the transmission delay, which is the time span between the emission of the first bit and the emission of the last bit of a frame. The transmission delay depends on the packet size and the network transmission rate. The propagation time, defined as the time between the emission of the first bit on the transmitter side and the reception of such a bit on the receiver side, also contributes to the end-to-end delay, although its impact for local area networks is negligible. Channel errors occurring during the transmissions of a packet may entail the retransmission of the corrupted packet. However, when dealing with real-time traffic, especially with periodic control packets in factory communication, the advantage of retransmissions has to be carefully evaluated, as there is a non-null probability that the retransmitted data becomes obsolete during its way to the destination, due to the generation of a ‘fresh’ value. However, a large part of the end-to-end delay at the application level is due to the *latency* at the *end nodes*. In [Ske02], with reference to switched Ethernet, it was demonstrated that the use of priorities for packets in the communication infrastructure is not on its own sufficient to guarantee that application-to-application deadlines will be met and that the concept of priority has to be extended to the end nodes in the protocol stack. Other works dealt with limitations of Switched Ethernet as far as real-time communication is concerned [Jas02]. In [LoB04] a significant reduction in the roundtrip delay for high priority traffic is obtained thanks to an approach based on both a prioritization mechanism in the protocol stack and multiple reception and transmission queues in the end nodes.

Many networked real-time systems feature applications with further requirements than time constraints only, for instance, mobility, dependability, composability, scalability, flexibility, security, safety. The way of dealing with real-time constraints in the presence of such requirements depends on the particular scenario.

2.4 Analytical Methods to assess performance of real-time networks

The analysis of real-time networks can be done with methods that have emerged either in the field of real-time systems or in the field of communication networks. Notable examples are RTA (Response Time Analysis) and Network Calculus.

RTA is a method to verify the schedulability of a task set in a processor or of a message set in a communication network, which is based on the response time computation and comparison with the deadline for each task or message in the set. If the deadlines are met for every instance of the tasks or messages that feature real-time constraints, then the schedulability is guaranteed. So, RTA provides an exact schedulability test. RTA has been firstly presented in [Jos86], but many evolutions of RTA then followed. In [Aus94] the restriction of the deadline being equal to the period was released and RTA was extended to tasks with deadlines less or equal to the period. [Leh90] extended RTA to deadlines greater than the period, thus enabling several active instances of the same messages. One of the most used results in industrial communications is the work where RTA was extended to CAN [Tin94], thus accounting for non-preemptive fixed priority scheduling. Using RTA, the queuing delay of a message can be obtained using a recursive equation. Following this result, several specific RTA-based analyses have been derived for different real-time networks, in the field of industrial automation. In [Alm02] a response-time analysis for both periodic, using fixed priorities, and aperiodic exchanges of variables (messages) is presented for WorldFIP, leading to a schedulability condition necessary and sufficient for the periodic traffic although just sufficient for the aperiodic one. A recent work by Bril [Bri06] revisited the RTA for ideal CAN network showing by means of examples with a high load (≈ 98%) that the analysis as presented in [Tin94] is optimistic. They proved that, assuming discrete scheduling, the problem can be resolved by applying the analysis for fixed-priority non-preemptive scheduling presented in [Geo96].

Network Calculus (NC) was introduced by Cruz in [Cru91a,Cru91b] to perform deterministic analysis in networks where the incoming traffic is not random but unknown, being limited by a known arrival curve. Network Calculus differs from queuing theory in the sense that it deals with worst case instead of average cases. The parameters of interest in this methodology are the delay, the buffer allocation requirements and the throughput. Using the arrival curve, often named α, and a service curve offered by the network, named β, and considering a traffic flow $R(t)$, it is possible to determine upper bounds for the backlog (the number of bits in "transit" in the system [Bou04]) and the delay.

Network calculus has been used in several real-time domains such as industrial automation, avionics and wireless sensor networks. Considering industrial automation, in [Geo04] Network Calculus was used to model a switched Ethernet architecture in order to evaluate the maximum end-to-end delays which are critical for industrial applications. In [Rid07] a stochastic network calculus approach to evaluate the distribution of end-to-end delays in the AFDX (avionics full duplex switched Ethernet, ARINC 664) network used in aircrafts such as Airbus A380 is presented. Network calculus has also been applied to the dimensioning of wireless sensor networks [Sch05].

3 Design Paradigms for Real-time Systems

3.1 Centralized vs. Distributed Architectures

A centralized architecture for a real-time system comes with the advantage that no communication between system components has to be considered, which saves cost and design effort. On the other hand, there are several reasons for building distributed real-time systems:

- Performance: A centralized system might not be able to provide all the necessary computations within the required timeframes. Thus, tasks are to be executed concurrently on networked hardware.
- Complexity: Even if it is possible to implement a real-time system with a centralized architecture, a distributed solution might come in with a lower complexity regarding scheduling decisions, separation of concerns, etc. In many cases, a complex monolithic system would be too complex in order to be verified and accepted as a dependable system.

- Fault tolerance: Ultra-dependable systems that are to be deployed in safety-critical applications are expected to have a mean-time-to-failure (MTTF) of better than 10-9 hours [Sur95]. Single components cannot provide this level of dependability; therefore, a system must be designed as a distributed system with redundant components in order to provide its service despite of particular faults.
- Instrumentation: Using a distributed sensor/actuator network allows to perform the instrumentation of sensors and actuators directly at the device. The interface to the particular devices is then implemented as a standardized digital network interface, avoiding possible noise pickup over long analog transmission lines and reducing the complexity in instrumentation [Elm03].

The distributed approach requires a well-suited communication system for connecting its different components. Such a communication system must provide hard-real time guarantees, a high level of dependability and means for fault isolation and diagnostics.

3.2 Composability and Scalability

In a distributed real-time system the nodes interact via the communication system to execute an overall application. An architecture supports *composability* if it follows from correct verification and testing of the components of the system (that is, its nodes and the communication system) that the overall system is correct. Think of a complex real-time system integrating components from many manufacturers. Each manufacturer does tests on the provided components. However, without supporting composability, the overall system cannot be proven to work correctly. Thus the integrated system has to be evaluated again, which comes with high effort and costs for each small change in its components. For real-time systems, composability in the value domain and composability in the time domain need to be ensured. An architecture supporting composability supports then a two-level design approach: The design of the components (possible done by particular manufacturers) and the overall system design do not influence each other as soon as the composability principle holds [Kop02].

Scalability of a real-time system also heavily depends on composability, since it requires new components not to interact with the stability of already established services If network resources are managed dynamically, it must be ascertained that after the integration of the new components even at the critical instant, i.e., when all nodes request the network resources at the same instant, the timeliness of all communication requests can be satisfied.

3.3 Time-triggered vs. Event-triggered Systems

There are two major design paradigms for constructing real-time systems, the *event-triggered* and the *time-triggered* approach. Event-triggered systems follow the principle of reaction on demand. In such systems the environment enforces temporal control onto the system in an unpredictable manner (interrupts). Conversely, time-triggered systems derive control from the global progression of time, thus the concept of time that appears in the problem statement is also used as basic mechanism for the solution.

The event-triggered approach is well-suited for sporadic actions and data, low-power sleep modes and best-effort soft real-time systems with high utilization of resources. Event-triggered systems can easily adapt to on-line changes and are flexible. However, they do not ideally cope with the demands for predictability, determinism and guaranteed latencies as a high analytical effort would be required to prove such properties.

Conversely, the time-triggered approach [Elm07] provides a low jitter for message transmission and task execution. The predictable communication scheme simplifies diagnosis of timing failures and the periodically transmitted messages enable a short and bounded error detection latency for timing and omission errors. The principle of resource adequacy guarantees the nominative message throughput independently of the network load. Moreover the time-

triggered paradigm avoids bus conflicts using a Time Division Multiple Access (TDMA) scheme, thus eliminating the need for an explicit bus arbitration. The downside of time-triggered systems is limited flexibility, worse average performance, the permanent full resource utilization and a higher effort (clock synchronization) in establishing the system. Support to on-line changes can be provided, but at the expense of efficiency (see Section 4.2). Most state-of-the-art safety-critical systems are based on a time-triggered approach. Furthermore, hybrid systems like the FlexRay bus [Fle05] or Time-Triggered Ethernet [Kop05] have gain attention in the last years.

3.4 Comparison of the real-time support provided by notable approaches

In order to preserve the real-time capabilities, real-time support in industrial networks has to be provided from the Medium Access Control (MAC) layer up to the application layer. There are several possible techniques to handle a shared medium. Table 1 lists some notable examples. A first approach is through controlled access protocols, which can be further classified as either centralized or distributed ones. The second approach is represented by uncontrolled access protocols (e.g., those based on the Carrier-Sense Multiple Access protocol), which can be extended with additional features to improve their real-time behaviour.

Table 1: Classifications of notable MAC layer protocols

Controlled access		Uncontrolled access
Centralized	Distributed	CSMA/CD (CSMA/Collision Detection)
Master/Slave	Token passing (virtual or physical)	CSMA/BA (CSMA/Collision Bitwise Arbitration)
	TDMA	CSMA/DCR (CSMA/ Deterministic Collision Resolution)
	Timed-Token	CSMA/CA (CSMA/Collision Avoidance)

In controlled access centralized methods, there is a central node acting as polling master that grants the stations the right to access to the channel. In this case, the timeliness of the transmissions on the network is up to such a central node, so it turns into a local scheduling problem at that node. Notable examples of these access methods in industrial networks are found in the Ethernet Powerlink [Eth07], Profibus DP [PRO07] and Bluetooth [IEE05] protocols. In the literature it has been shown that the use of a real-time scheduling algorithm (instead of a non-real-time one such as round-robin or FIFO algorithms) combined with a proper timing analysis makes it possible to compute the response time of real-time messages and avoid deadline miss [Tov99a,Col07]. In controlled access distributed protocols, transmissions are ruled according to a distributed mechanism, involving either the circulation of a physical token, or a virtual token passing, or a time division multiple access policy which allocates transmission instants to each node. In these cases, to enforce a timely behaviour on the network and thus to achieve transmission timeliness, suitable timers and policies [Mal95,Tov99b] or a global synchronization framework [Kop97] are required, respectively. Notable examples of networks provided with these access methods are Profibus [ES96], P-NET [ES96], TTP/C [Kop97], TTE [Kop05]. Conversely, in uncontrolled protocols, such as CSMA-based ones, each station decides autonomously when to transmit, obeying only to the protocol-specific rules. When collisions may occur, like in the CSMA/CD protocol, the one used by the Ethernet, achieving predictability

becomes a issue and suitable techniques, such as traffic smoothing, have to be adopted to bound the medium access time [Kwe04,LoB05]. A different situation arises with the CSMA/BA, where the collisions are non-destructive for the highest priority message, which then goes through without delay. CSMA/BA is used in the Controller Area Network (CAN) bus and several studies exist which show that it is possible to calculate the response time of real-time messages over the CAN bus [Tin94,Bri06]. In CSMA/CA collisions may occur, thus the medium access time is non-predictable. The highest probability of a collision exists just after the medium becomes idle following a busy medium, because multiple stations could have been waiting for the medium to become available again.

4 Design challenges in real-time industrial communication systems

4.1 Real-time and Security

Real-time industrial communication systems can be subject to different security threats. Such attacks can be message integrity, fake messages, intrusion, impersonation, denial of service (DoS), etc. DoS attacks can be difficult to handle due to the constrained resources available in the embedded networks used to support automation applications. DoS attacks can be handled either by prevention or by detection. A common requirement for preventing DoS attacks is the possibility to limit physical access to the network. Detection is typically done by observing the network behavior to detect uncommon situations and, in case of detection, by acting with combat measures. In [Gra08] a solution combining detection and prevention is proposed for automation networks. As industrial communications often deal with safety critical real-time systems, security services must also be taken into consideration from the point of view of efficient resource utilization. A balanced harmonization among dependability, security and real-time aspects must then be performed. In a failure situation, when resources become scarce, one must prioritize certain applications/users/services over others to be able to retain the most critical applications [Fal08]. Different dependability, security and/or real-time requirements will lead to different solutions and, for the most demanding services, the supporting mechanisms could be the most expensive ones, at least in terms of resource usage.

4.2 Real-time and Flexibility

The specification of the requirements of real-time systems is a task that can be hard due to incomplete knowledge of the application and of the dynamic scenarios where the systems will operate. If the requirements are not fully known before runtime or if the operating conditions change during operation or if it is required to add or remove system components during operation, then flexibility is an important issue. For industrial control systems, Zoitl [Zoi09] discusses an approach for reconfigurable real-time applications based on the real-time execution models of the new International Electrotechnical Commission's (IEC) family of standards (IEC 61499).

At the level of the communication protocol, different approaches for flexibility can be identified. As discussed previously, event-triggered communication systems such as the ones based on CAN can react promptly to communication requests that can be issued at any instant in time, letting the bus available in the absence of events to communicate. On the other side, time-triggered systems such as TTP [TTA03] are less flexible because communication must take place at predefined instants, being defined at pre-runtime. Protocols such as TTP allow mode changes among predefined static modes or allow reservation of empty slots to include messages which transmission is decided on line, but this wastes bandwidth, as the slots are reserved even if they are not used.

In spite of this division, a number of hybrid protocols has been defined in the last 20 years. The most prominent hybrid protocols are ARINC-629 [AEE94] from the avionics domain, FlexRay [Fle05] from the automotive field as well as Time-Triggered Ethernet (TTE) [Kop05]. Both these protocols enable on-line scheduling using FTDMA that is a flexible TDMA. Another example is FTT-CAN [Alm02] which uses a time-triggered approach where periodic traffic is conveyed with the control of a centralized dispatcher coexisting with "legacy" event-triggered CAN traffic transmitted in a specific window.

While TDMA is in general a static scheme, extensions to make it more flexible have been proposed. For example, P-NET fieldbus [ES96] uses a virtual token passing approach that enables a node to transmit earlier if the previous slot was not used to transmit a stream. This technique was ported to Ethernet and called VTPE, virtual token passing Ethernet [Car03].

4.3 Offering real-time support to wireless communication

In wireless industrial networks, due to the error-prone nature of the wireless channel, the deterministic view on schedulability of a set of real-time streams is no longer applicable and it should be replaced by a probabilistic view, based on the probability of fulfilling some industrial-related QoS figures. Such figures may take packet losses into account as well. Under this perspective, a set of real-time streams is considered to be schedulable (for fixed assumptions on the wireless channel error process) when all of its streams achieve a long-term success probability of, for example, 99% [Wil08]. To assess the schedulability of real-time traffic flows in wireless industrial networks, suitable analysis methodologies are needed. Promising approaches are represented by stochastic models, such as finite-state Markov channels (FSMC) [Ara03,Bab00,Has04], in which the signal strength at a receiver is varied according to a Markov chain with a finite number of states, and stochastic network calculus, in which deterministic end-to-end time bounds are replaced by stochastic ones [Jia06].

References

[AEE94] Airlines Electronic Engineering Committee, "ARINC Specification 629-3: IMA Multitransmitter Databus", 1994

[Alm02] L.Almeida, P. Pedreiras, J.A. Fonseca, "The FTT_CAN protocol: why and how", IEEE Transactions on Industrial Electronics, 49, 2002.

[Ara03] J. Arauz and P. Krishnamurthy, "Markov modeling of 802.11 channels," in Proc. 58th IEEE Vehicular Technology Conf. (VTC) 2003-Fall, Oct. 2003, pp. 771–775.

[Aud91] N. C. Audsley, A. Burns, M. F. Richardson, A. J. Wellings, „Real-Time Scheduling: The Deadline Monotonic Approach", Proc. 8th IEEE Workshop on Real-Time Operating Systems and Software, 1991.

[Bab00] F. Babich, O. E. Kelly, and G. Lombardi, "Generalized Markov modeling for flat fading," IEEE Trans. Commun., vol. 48, no. 4, pp. 547–551, Apr. 2000.

[Bon03] A. Bonaccorsi, L. Lo Bello, O. Mirabella, P. Neumann, A. Pöschmann, "A Distributed Approach to Achieve Predictable Ethernet Access Control in Industrial Environments", Proc. of the IFAC International Conference on Fieldbus Systems and their Applications (FET'03), July 2003.

[Bou04] J-L. Le Boudec, P. Thiran "Network Calculus", Lecture Notes in Computer Sciences vol. 2050, Springer Verlag, 2001.

[Bri06] R. J. Bril, J. J. Lukkien, R. I. Davis, A. Burns, "Message response time analysis for ideal controller area network (CAN) refuted", the 5th Int. Worshop on Real-Time Networks, Satellite Workshop of ECRTS 06, Dresden, Germany, July 4, 2006.

[Car03] F. B. Carreiro, J. A. Gouveia Fonseca, P. Pedreiras, "Virtual Token-Passing Ethernet – VTPE", in Proc. of the 5th IFAC International Conference on Fieldbus Systems and their Applications (FeT'2003), Aveiro, Portugal, 2003.

[Col07] M. Collotta, L Lo Bello, O. Mirabella, “Deadline-Aware Scheduling Policies for Bluetooth Networks in Industrial Communications”. In Proc. of the IEEE Second International Symposium on Industrial Embedded Systems - SIES'2007, Lisbon, Portugal, 4-6 July 2007, pages 156–163.

[Cru91a] R. L. Cruz, “A calculus for network delay, part I: Network elements in isolation,” *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 114–131, Jan. 1991.

[Cru91b] R. L. Cruz, “A calculus for network delay, part II: Network analysis,” *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 132–141, Jan. 1991.

[Dem90] A. Demers, S. Keshav, S. Shenker, “Analysis and simulation of a fair queuing algorithm“, Journal of Internetwoking Research and Experience, Oct.1990.

[Elm03] W. Elmenreich and S. Pitzek. Smart Transducers - Principles, Communications, and Configuration. In Proc. of the 7th IEEE International Conference on Intelligent Engineering Systems, volume 2, pages 510-515, Assuit - Luxor, Egypt, Mar. 2003.

[Elm07] W. Elmenreich. Time-triggered Transducer Networks. Habilitation thesis, Vienna University of Technology, 2007.

[Elm09] W. Elmenreich, Editor. Embedded Systems Engineering. Vienna University of Technology, Vienna, Austria, 2009, ISBN 978-3-902463-08-1.

[ES96]European Standard EN 50170. Fieldbus. Volume 1: P-Net, Vol. 2: PROFIBUS, Vol. 3: WorldFIP, 1996.

[Eth07] Ethernet Powerlink Standard. included in IEC 61158-2, Edition 4.0, 2007

[Fal08] L. Falai et al, Mechanisms to provide strict dependability and real-time requirements, EU FP6 IST project HIDENETS, deliverable D3.3, June 2008.

[Fle05] Flexray Consortium. FlexRay Communications System Protocol Specification Version 2.1, 2005. Available at http://www.flexray.com.

[Geo04] J.-P. Georges, T. Divoux, E. Rondeau, “A formal method to guarantee a deterministic behaviour of switched Ethernet networks for time-critical applications”, IEEE Intemational Symposium on Computer Aided Control Systems Design, Taipei, Taiwan, September 2-4, 2004.

[Gra08] W. Granzer, C. Reinisch, and W. Kastner. Denial-of-Service in Automation Systems. In Proc. of the IEEE Conference on Emerging Technolgiees and Factory Automation, 2008.

[Geo96] L. George, N. Rivierre, and M. Spuri. Preemptive and nonpreemptive real-time uni-processor scheduling. Technical Report 2966, Institut National de Recherche et Informatique et en Automatique (INRIA), France, September 1996.

[Has04] M. Hassan, M. M. Krunz, and I. Matta, “Markov-based channel characterization for tractable performance analysis in wireless packet networks,” IEEE Trans. Wireless Commun., vol. 3, no. 3, pp. 821–831, May 2004.

[IEE05] IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 15.1: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for wireless personal area networks (WPANs), IEEE Computer Society— Sponsored by the LAN/MAN Standards Committee, 2005.

[Jas02] J. Jasperneite, P. Neumann, Theis M., Watson, K. “Deterministic Real-Time Communication with Switched Ethernet”, Proc. ofWFCS’02, 4Th IEEE Workshop on factory Communications Systems, pp.11-18, vasteras, Sweden, Aug.2002.

[Jia06] Y. Jiang, “A basic stochastic network calculus,” in Proc. ACM SIGCOMM Conf. Network Architectures and Protocols, Pisa, Italy, Sep.2006.

[Jos86] M. Joseph and P. Pandya. Finding response times in a real-time system. The Computer Journal, 29(5):390–395, 1986.

[Kop02] H. Kopetz and R. Obermaisser. Temporal Composability. IEE’s Computing & Control Engineering Journal, 13(4):156–162, Aug. 2002.

[Kop05] H. Kopetz, A. Ademaj, P. Grillinger, and K. Steinhammer. The Time-Triggered Ethernet (TTE) Design. In Proc. of the 8th International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC), pages 22–33, Seattle, WA, USA, May 2005.

[Kop97] H. Kopetz. Real-Time Systems, Design Principles for Distributed Embedded Applications. Kluwer Academic Publishers, Boston, Dordrecht, London, 1997.

[Kwe03] S. K. Kweon.and K.G. Shin (2003). Statistical Real-Time Communication over Ethernet. In: IEEE Transactions on Parallel and Distributed Systems,14(3), pp. 322–335.

[Kwe04] S. K. Kweon, M.G. Cho and K.G. Shin (2004). Soft Real-Time communication over Ethernet with adaptive traffic smoothing. In: IEEE Transactions on Parallel and Distributed Systems,15(10), pp. 946–959.

[Leh90] J. Lehoczky. Fixed priority scheduling of periodic task sets with arbitrary deadlines. In *Proc.* 11*th IEEE Real-Time Systems Symposium (RTSS)*, pp. 201–209, December 1990.

[Liu00] J.W.S. Liu, Real-Time Systems, Prentice Hall, 2000.

[Liu73] L. Liu and J. Layland. "Scheduling algorithms for Multiprogramming in a Hard Real-Time Environment", Journal of ACM, Vol. 20, No. 1, 1973, pp. 46-61.

[LoB05] L. Lo Bello,G. Kaczynski, O. Mirabella, "Improving the Real-Time Behaviour of Ethernet Networks Using Traffic Smoothing", IEEE Transactions on Industrial Informatics, Special Section on Factory Communication Systems, Vol.1, N.3, pp. 151-161, ISSN: 1551-3203, IEEE Industrial Electronics Society,Piscataway, USA, Aug. 2005.

[LoB06]L. Lo Bello, G. Kaczynski, F. Sgro', O. Mirabella, "A wireless traffic smoother for soft real-time communications over IEEE 802.11", in Proc. of ETFA06, the 11th IEEE International Conference on Emerging Technologies and Factory Automation, Sept. 2006, pp. 1073-1079.

[Mal95]N. Malcom, W. Zhao, "Hard real-time communications in multiple access networks, R*eal-Time Systems*, 9, 75-107, 1995.

[Mor08] R.Moraes, P. Portugal, F. Vasques, Fonseca J "Limitations of the IEEE 802.11e EDCA protocol when supporting real-time communication," in Proc. of the 7th IEEE International Workshop on Factory Communication Systems (WFCS'08), Dresden, Germany, 2008.

[Ped05] 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 Transactions on Industrial Informatics, 1(3):162{172, 2005.

[PRO07] PROFIBUS Standard – DP Specification. included in IEC 61784-3-3, Edition 1.0, 2007

[Ray07] M. Raya and J. P. Hubaux. Securing Vehicular Ad Hoc Networks. Journal of Computer Security, Special Issue on Security of Ad Hoc and Sensor Networks, Vol. 15, pp. 39 - 68, 2007.

[Rid07] F. Ridouard, J.-L. Scharbarg, C. Fraboul "Stochastic network calculus for end-to-end delays distribution evaluation on an avionics switched ethernet", Proc. 5th International Conference on Industrial Informatics (INDIN 2007), Vienna, Austria.

[Sch05] J. Schmitt, U. Roedig, "Sensor Network Calculus – A framework for worst case analysis", Proc. of the International Conference on Distributed Computing in Sensor Systems (DCOSS05), Marina del Rey, USA, 2005

[Sha04] L. Sha, T. Abdelzaher, K. Arzen, A. Cervin, T. Baker, A. Burns, G. Buttazzo, M. Caccamo, J. Lehoczky, A. Mok, "Real-Time Scheduling Theory: A Hystorical Perspective", Real-Time Systems, Vol. 28, Issue 2-3, pp. 101-155, December 2004.

[Ske02] T. Skeie, S. Johannessen, and Ø. Holmeide. The Road to an End-to-End Deterministic Ethernet, In: Proc. of 4th IEEE International Workshop on Factory Communication Systems (WFCS), 2002.

[Sta90] J. Stankovic, K. Ramamrithan. What is predictability for real-time systems? Journal of Real-Time Systems, 2, 1990.

[Stan88] J. Stankovic, K. Ramamrithan, editors. Tutorial on Hard Real-Time Systems, IEEE Computer Society Press, Washington, D.C, 1988.

[Sur95] N. Suri, C. J. Walter, and M. M. Hugue, editors. Advances in Ultra-Dependable Systems. IEEE Press, 1995.

[TTA03] TTAGroup. Specification of the TTP/C Protocol V1.1, 2003. Available at http://www.ttagroup.org.

[Tin94] K. Tindell, H. Hansson, and A. Wellings. Analysing real time communications: Controller area network (CAN). In *Proc.* 15*th IEEE Real-Time Systems Symposium (RTSS)*, pp.259–263, Dec. 1994.

[Tov99a] E. Tovar, “Supporting real-time communications with standard factory- floor networks”, Ph.D. dissertation, Dept. Elect. Eng., Univ. Porto, Porto, Portugal, 1999.

[Tov99b] E. Tovar, F. Vasques, “Cycle time properties of the PROFIBUS timed token protocol”, Computer Communications, 22, 1206-1216, 1999.

[Wil08] A. Willig, “Recent and Emerging Topics in Wireless Industrial Communications: A Selection”, IEEE Transactions on Industrial Informatics, Vol. 4, No. 2, May 2008, pp.102-124.

[Zoi09] A. Zoitl, "Real-time execution for IEC 61499", International Society of Automation (ISA), 2009