Performance Analysis of Producer/Consumer Protocols over IEEE802.11 Wireless Links

Daniele Miorandi [*†] and Stefano Vitturi [*†]
[*] Department of Information Engineering, University of Padova,
via Gradenigo 6/B, 35131 Padova (Italy).
Email:{daniele.miorandi,vitturi}@dei.unipd.it
[†] Italian National Council of Research, IEIIT–CNR

Abstract

Thanks to recent technological advances, wireless networks are beginning to represent an interesting opportunity for factory communication systems. Among the off–the–shelves solutions for radio communications, the IEEE802.11 technology is one of the most promising. However, industrial applications typically impose severe requirements in terms of both real–time performances and dependability. In this paper we consider one of the most popular models of fieldbus protocols, namely the Producer–Consumer, and study the possibility of implementing it on top of IEEE802.11 protocol suite. After a description of how the Producer–Consumer services could be mapped onto IEEE802.11, we introduce an analytical model, which enables us to evaluate two important performance indexes: the update time jitter and the mean alarm latency. The analysis is validated by means of numerical simulations.

1 Introduction

Recently, we assisted to an impressive growth of wireless communication systems in several application fields. Among the positive effects of such a growth we have both an increased dependability and a cost reduction, thanks to the arising of scale economies. Moreover, some performance figures of wireless systems (e.g. the transmission rate) are becoming comparable with those of wired networks. Consequently, wireless networks are beginning to represent a viable choice also for industrial communication systems, where we are often faced with strong requirements in terms of real time communications.

There are several benefits deriving from such a choice: in principle, it allows to connect moving equipment which, otherwise, would be doomed to remain isolated. Moreover, wireless links may be used to replace (at least in some specific applications) the traditional wired connections, eliminating the cabling which may result expensive and often difficult to deploy.

Thus, it is likely that, in the near future, we will assist to a considerable adoption of wireless networks in factory communication systems.

Although several products are already available for radio communications, only a few of them are potentially suitable for industrial applications.

An important analysis on this issue was made by the R–FIELDBUS project [1], supported by the European Commission in the 5th FP. The R–FIELDBUS project was aimed at the design of a complete wireless fieldbus, including also a stack for multimedia industrial communication. A very interesting review has been carried out in order to compare the physical layers of some available products. As a result, the Direct Sequence Spread Spectrum (DSSS) physical layer specified by the IEEE802.11 [2] committee has been selected as suitable for industrial applications.

The use of IEEE802.11 in industrial environments has been considered also in the scientific literature: in [3] some specific measurements under different operating conditions are performed on the physical layer; in [4] a modification of the Profibus FMS [5] MAC protocol is proposed to obtain the best performances from a wireless version of this fieldbus based on IEEE802.11.

Obviously, also the upper layers have a relevant impact on system performance. In particular, it has to be considered that, at the lowest levels of factory communication systems, both cyclic and acyclic real time data exchange are typically required. For this reason, the protocols used by field networks are substantially different from those of LANs. For example, the R–FIELDBUS has adopted the Profibus DP protocol [6] for real time communication, whereas the multimedia industrial communication is implemented via the TCP/IP suite [7] [8].

In general, the specific protocols used by the fieldbuses have to accomplish two different tasks, typical of industrial applications: the cyclic update of process variables and the handling of acyclic events, such as those generated by alarms. Both types of traffic may actually take place at a very high rate.

The protocols used to handle such operations are basically of two types: Master–Slave and Producer–Consumer.

Master–Slave, of which typical examples are Profibus DP and Interbus [9], are based on a point–to–point data exchange implemented by the action of master devices which poll the slaves by means of suitable communication services. Producer–Consumer protocols are designed for the exchange of a set of identified variables. For each variable only a producer node exists, whereas several nodes may consume it. Fieldbuses using such a type of protocol are, for example, WorldFIP [10] and the original IEC fieldbus project, that has been standardized as a Technical Specification [11].

For the sake of clarity, it has to be observed that Producer–Consumer protocols are actually placed at the data link layer, whereas Master–Slave protocols are typically implemented as applications which use the services offered by the data link layer (for example, Profibus DP defines a set of functions which are mapped on the data link layer services). However, from a functional point of view, in most cases, both Master–Slave and Producer–Consumer may be considered equivalent, since they supply the services necessary to handle both cyclic and acyclic data traffic.

In this paper we focus on the behavior of the Producer–Consumer protocols when implemented on top of the IEEE 802.11 protocol stack. More in detail, we will consider a possible model of a wireless fieldbus, based on a Producer–Consumer architecture. We will supply details on how the protocol services could be mapped onto the IEEE 802.11 and we will analyze its performances.

The proposed protocol is designed to handle only real–time data, since this is the typical requirement of industrial applications. For this reason, in our work we will not consider any different type of traffic.

The remainder of this paper is organized as follows. Sections 2 deals with the characteristics of the IEEE 802.11 wireless LAN. Section 3 describes the model of the Producer–Consumer protocol we adopted. Section 4 provides a theoretical framework for performance evaluation of the proposed protocol, together with the results obtained analytically. Some of these results have also been validated by means of software simulations. Section 5 concludes the paper.

2 Background: the IEEE802.11 wireless LAN

IEEE802.11 [12] is the de facto standard for wireless local area networks (WLANs).

As all other member of the IEEE 802.x family, 802.11 specifies the characteristics of both the physical (PHY) and medium access control (MAC) layers. At the physical layer, the standard encompasses three different options: infrared, frequency hopping spread spectrum and direct sequence spread spectrum. We focus on the last option (the one currently implemented), and, furthermore, use the parameters described for operations in the 2.4 GHz ISM band, known as 802.11b [13], which is able to provide data rates up to 11 Mb/s. In order to adapt the capabilities of the physical medium dependent system to the PHY services, a physical layer convergence protocol (PLCP) is provided, which acts as interface with the MAC layer.

In IEEE802.11 standard, the medium access control is based on a distributed CSMA/CA mechanism. A node listens to the channel for a time equal to the distributed inter–frame spacing (DIFS). If the medium is sensed idle, then a random backoff is generated. During the backoff the node keeps on sensing the channel. If, at the end of the backoff, the medium is still idle, the node starts transmitting. Since no channel load sensing mechanism is provided, an explicit acknowledgment is necessary to inform the node of the success/failure of its transmission. To accomplish that, when a node receives a packet, after a short interframe spacing (SIFS), it sends a short ACK packet to inform the source of the outcome of the previous transmission. Since collisions may occur, a truncated binary exponential backoff scheme is provided to resolve contention for the channel. At each transmission attempt, the length of the backoff interval, expressed in slots, is randomly chosen in the set $\{0, 1, \ldots, CW - 1\}$, where CW denotes the actual contention window size. At the beginning, CW is set to a predefined value CW_{min}. If a collision or loss occurs, the value of CW is doubled and another transmission attempt is made. The contention window cannot grow indefinitely, but may reach a maximum value of $2^{m'}CW_{min}$; moreover, if a packet incurs m collisions, where $m \geq m'$, it is dropped. If a transmission is detected while the backoff counter has not reached zero, its value is frozen and reloaded as soon as the channel is sensed idle again for a DIFS.

The procedure previously described, known as basic access, suffers (as all CSMA—based MAC protocols) from the well–known hidden–terminal problem. Thus, an optional RTS/CTS mechanism is encompassed by the standard. In this case, after a DIFS and a random backoff, a RTS packet is sent to the intended destination. After a SIFS, the destination replies with a CTS, signalling to all the stations in range the foreseen duration of the packet exchange. After a SIFS, the sender may thus start its transmission. Note that, in this way, a virtual channel sensing mechanism is provided, since all the stations which received the CTS may update their network allocation vector (NAV) and go in stand–by for the whole duration of the packet exchange (the channel is "virtually" sensed busy). The decision whether to use basic access or the RTS/CTS mechanism is made, according to the standard, on the basis of the packet length. If the packet to be transmitted is longer than a given threshold, then RTS/CTS is used; otherwise, the MAC entity proceeds according to the basic access mechanism.

The standard provides also a way of broadcasting messages, in order to offer asynchronous unacknowledged services; furthermore, for such messages, only the basic access can be used. Notice that there is no MAC–layer recovery on broadcast messages, which thus turn out to suffer higher losses in error–prone channels.

What we described above, is the so–called Distributed Coordination Function (DCF) operation mode; the standard encompasses also an optional Point Coordination Function (PCF) mode, which is well suited to centralized operation and the handling of delay–sensitive applications. However, most of the WLAN cards actually available on the market do not implement PCF for complexity reasons. For such a reason we will not take into consideration the PCF mode in the proposed protocol.

Finally, the IEEE802.11b MAC provides a rate adaptation mechanism which is aimed at coping with varying channel conditions. Three different modulation schemes are provided (BPSK, QPSK and CCK), which enable a set of four different transmission range, 1, 2, 5.5 and 11 Mb/s. According to the protocol, the choice of the modulation scheme to be employed is based on an estimate of the channel conditions, in order to keep a low probability of packet losses. Indeed, the ARQ mechanism provided at the MAC layer is designed to cope with collisions and not with channel errors. In the following, we will assume that packet losses are rare events, so that the rate adaptation mechanism does not react and keep a data rate R_{data} of 11 Mb/s. Throughout the paper, we assume that the 11 Mb/s transmission rate belongs to the BSS basic rate set [13], so that all packets, including ACKs and broadcast, are transmitted at the data rate[1].

3 The Producer–Consumer protocol over the IEEE802.11 wireless LAN

3.1 General

The IEEE802.11 standard specifies the use of the Logical Link Control, LLC [14] as interface towards the upper layers. Hence, a suitable communication profile of the architecture we are proposing is shown in Fig. 1.

LLC offers three types of services to the upper layers, namely: connectionless acknowledged services, connection oriented unacknowledged services and connectionless unacknowledged services. Thus, the communication functions of the Producer–Consumer protocol may be implemented by means of the most suitable LLC services.

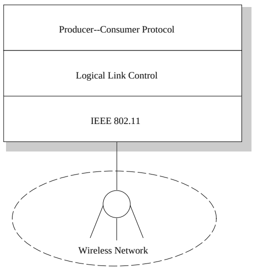


Figure 1. Communication profile of the proposed architecture.

3.2 The protocol model

A Producer–Consumer protocol is based on the exchange of identified variables which are produced in reply to production requests issued by a node. Although, in general, such requests could be generated by any node, in practice the most popular protocols allow only a selected node to perform such a function. This role is played, for example, by the Bus Arbiter in WorldFIP, or by the Link Active Scheduler, (LAS) in IEC fieldbus. Consequently, we will maintain such a feature in the protocol model, referring to such a node as the scheduler.

It is worth mentioning that the model we propose, although placed on top of LLC, represents actually a data link layer protocol very similar to those used by both WorldFIP and the IEC fieldbus. In order to implement an application layer protocol on top of the proposed Producer–Consumer model, the most obvious choice would be to consider the protocols already adopted by the above mentioned fieldbuses. Thus, for example, a WorldFIP–like implementation would require of realizing a set of services such those defined by the Manufacturing Periodical/aperiodical Services (MPS) module of the WorldFIP application layer. However, also different approaches could be adopted. For example, we could refer to both the CANopen [15] and DeviceNet [16] application layer protocols which are based on the definition of a set of communication objects that may be exchanged on the network. Also in this case the implementation should be straightforward, since both CANopen and DeviceNet are defined on top of the Controller Area

[1] In principle, broadcast messages, ACKs and RTS/CTS exchange are performed at a lower data rate, usually 1 Mb/s, to ensure that all hosts are able to receive them. However, many cards available on the market transmit all packets at 11 Mb/s, as we assume.

Network [17] data link layer, which is based actually on a Producer–Consumer protocol.

With the proposed model, the cyclic exchange of data is achieved by periodically programming in the scheduler the production requests of the variables. As a consequence, the producer nodes will transmit on the network the variables with the specified period.

Such a behavior suggests that the most suitable LLC services for implementing the cyclic data exchange are the unacknowledged connectionless services. An example of the sequence request–production is shown in Fig. 2. As can be seen, the production request issued by the scheduler is mapped on a LLC service which causes the broadcasting of an IEEE802.11 PDU to the network nodes. As a result, only the actual producer of the variable (node 2 in the example) accesses the network to transmit its value, which may be acquired by all the consumers (nodes 1 and n in the example).

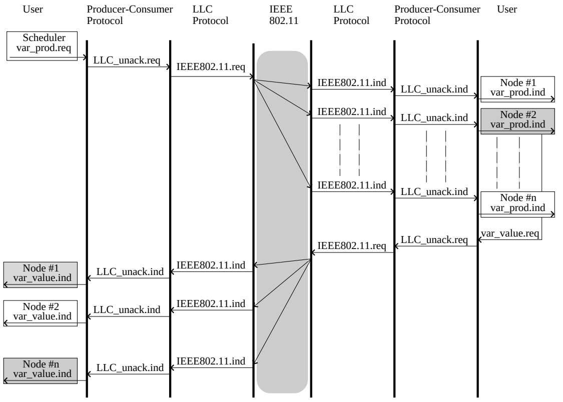


Figure 2. Example of a production request.

Acyclic activities take place in the time intervals between subsequent scheduled operations. A node requested to produce a variable may signal, at the same time, the necessity of sending/receiving acyclic data. Consequently, the scheduler in the immediately available time interval will satisfy the request of that station. More in detail, in WorldFIP, if the station requested the production of a variable, the scheduler will issue the appropriate request. Conversely, if the station needs to directly send data to other network partners, then the scheduler will grant that station with the right to access the network.

As a result, the traffic in the network is characterized by the repetition of "macrocycles" containing cyclic requests interleaved by acyclic operations.

A similar procedure is adopted by the IEC fieldbus: in this case, the scheduler simply passes a delegated token to the station that signalled the need of acyclic activities.

In this paper we will use this latter method for the Producer–Consumer protocol we are analyzing.

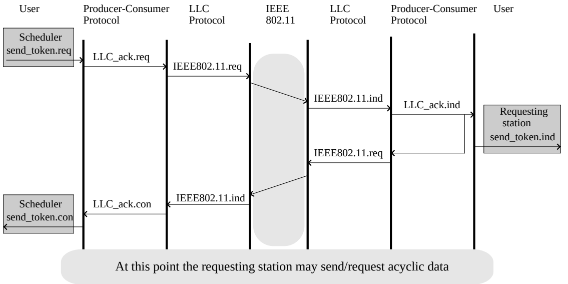


Figure 3. Example of acyclic data transmission.

Moreover, in order to ensure that acyclic messages arrive correctly at destination, we implement them by means of LLC confirmed services. An example of such a procedure is shown in Fig. 3. As can be seen, the scheduler sends the delegated token to the requesting station with the primitive "snd_token.req". The request is mapped onto a LLC confirmed service and sent to the destination. The Producer–Consumer protocol of the requesting station receives the delegated token and contemporaneously confirms the correct reception. When the confirm primitive arrives at the scheduler, the acyclic transmission may start.

However, it should be evident that such a procedure presents some drawbacks. In order to maintain the refresh rate at an acceptable value for cyclic data exchange, the scheduler cannot, in principle, prevent cyclic and acyclic operations to overlap. This means that, while the network is processing an acyclic data request by a device, the scheduler may attempt sending a production request for the next variable. Such a situation may be encountered since the retransmissions, which can always occur when acknowledged services are used, prolong the duration of the acyclic data services. Thus, as can be readily understood, some collision problems may arise. For this, however, we rely on the ability of the MAC protocol to avoid collisions. In any case, even if such a situation takes place, this can have a detrimental impact on broadcast messages only, which are not going to be retransmitted. The overall effect of a sporadic missing of a cyclic update of a variable operation will not, in most cases, cause instability problems in the system behavior. On the other hand, acyclic operations (which may carry critical data) are highly likely to be successfully performed.

Finally, we assume that the scheduler, when receiving an acyclic request, delivers the delegated token to the requiring station immediately after the end of the current cyclic operation. Equivalently, the interval between any two subsequent cyclic operations within a macrocycle is supposed to accommodate (in absence of retransmissions) at least one acyclic operation. It is clear that in this way we limit the maximum refresh rate of cyclic activities. However, it should be also evident that this can represent a problem only in presence of very time critical applications. Hence, since the major objective of the paper is to study the implementation of a Producer–Consumer protocol on top of IEEE802.11 wireless links, the above assumption does not seem to represent a limiting factor for our work.

The resulting automaton that describes the operation of the scheduler is shown in Fig. 4. As can be noted, when the scheduler is in the state "Cyclic", only such a type of activity is carried out on the network; with the event "pass token" the model enters the state "Acyclic" where network access is granted by the scheduler to another station for acyclic activities. The state "Both" is necessary to describe the aforementioned situation in which the retransmissions cause a collision with the next scheduled cyclic activity.

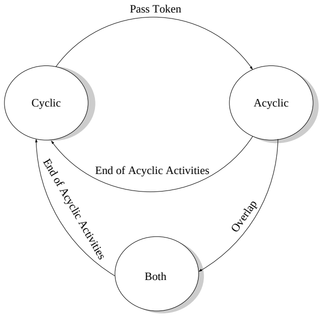


Figure 4. Model of the Producer–Consumer protocol.

4 Performance Analysis

In this section we present a performance analysis of the Producer–Consumer protocol when running over an IEEE802.11 wireless network. In particular, we will focus on two metric of interest: namely the jitter which may affect the cyclic update of variables and the mean alarm latency.

The update period of a variable, T_u, is defined as the period with which that variable is correctly received: since transmission errors may occur during the production phase, T_u is a random variable. Hence, the update period jitter may be effectively described by the variance of T_u.

The alarm latency, denoted by D, is defined as the time encompassed between the generation, by a device, of an alarm message and its successful transmission to the intended destination.

In particular, we work under the following assumptions:

(i) all packets have the same length L (in bits) and they are short enough, so that basic access is always used [2];

(ii) alarm messages at the various devices are generated according to independent Poisson processes of intensity λ;

(iii) channel errors are independent; furthermore, they represent rare events, so that the rate adaptation mechanism of 802.11 does not react and keep on transmitting at the highest possible rate (in our case 11 Mb/s);

(iv) no packet drops take place for acknowledged services.

Note that, according to assumption (iii), the recovery of packet losses is addressed by the MAC retransmission mechanism. This means that, since errors are assumed to be independent, a packet would be dropped with probability $P_{drop} = P_e^m$, where P_e is the packet error probability. For an IEEE802.11b network, using the basic access, we have $m = m' = 4$. Thus, even with $P_e = 10^{-1}$, we would have $P_{drop} = 10^{-4}$, which shows the soundness of assumption (iv). However, in the case of broadcast packets, the probability of loosing a packet is $P_{loss} = P_e$, since no retransmission mechanism can be invoked. It is worth recalling that, according to the results in [3], a channel model with independent errors could not reflect the IEEE802.11 behavior in an industrial plant. The analysis of this case is then more complex, and it is deferred to a future work.

In the rest of the paper, we will use the following notation: for a random variable X, we will denote its mean by $x = E[X]$, and its variance by $\sigma_x^2 = E[X^2] - x^2$, where $E[\cdot]$ denotes the statistical expectation operator. Let us denote by T_{ack} and T_{nack} the transmission time of a packet sent with an acknowledged or unacknowledged service, respectively.

For the sake of clarity, we assume that the probability of error is the same on every link; however, the model can be easily adapted to account for different P_e on the various links (similarly, also different alarm generation rates can be accounted for). Further, with a slight abuse of notation, we denote by T_r the (deterministic) refresh time of the variable of interest. More precisely, T_r represents the interval between two successive production requests of the variable under study, so that we refer to T_r as the "refresh cycle". No restrictions are posed on the number of productions of a variable within a macrocycle. Without loss

[2] This is reasonable since, in most cases, only a few octets of data have to be exchanged between devices in a fieldbus. Further, the "packet" is intended as LLC PDU.

of generality, we assume that each device generates one and only one variable (if a device generates more variables, we can introduce some fictitious ”virtual devices” and proceed accordingly).

In Producer–Consumer protocols, unacknowledged services are used to perform the cyclic data exchange. Then, the transmission time of a variable (and of the variable production request) is given by T_{nack}. For the sake of simplicity, we do not consider the possible collisions of a production request with acyclic data services. However, we have to account for the possible loss of a packet due to channel errors.

Let us consider, without loss of generality, the first variable to be generated. Its value is updated when the following two events take place:

1. the producer correctly receives the production request issued by the scheduler;

2. the produced variable is correctly received by the intended destination(s).

Since the variable is produced at each refresh interval, the update time may be thought as the time to the first success in a Bernoulli trials process, where each trial has a success probability $(1 - P_e)^2$. Each trial has a (deterministic) duration T_r, so that we obtain:

$$T_u = k \cdot T_r, \tag{1}$$

where k is a geometric random variable, having probability mass function:

$$p_k(n) = (1 - P_e)^2 \cdot \left[1 - (1 - P_e)^2\right]^{(n-1)}, \ n = 1, \dots \tag{2}$$

Then, the mean update time is given by:

$$t_u = \frac{T_r}{(1 - P_e)^2}. \tag{3}$$

Similarly, the update time variance (jitter) is obtained as:

$$\sigma^2_{T_u} = (T_r)^2 \cdot \frac{1 - (1 - P_e)^2}{(1 - P_e)^4}. \tag{4}$$

Notice that, in the equation above, we neglected the randomness due to the backoff procedure; simulation results will show the soundness of our approximation.

¿From a network dimensioning point of view, we may be interested in providing a given reliability degree. This may be expressed in terms of outage probability, where the outage event corresponds, for example, to $\{T_u > \Psi\}$, where Ψ is a threshold value which depends on the application of interest. Thus, we may write:

$$P[T_u > \Psi] = \sum_{n=\lfloor \frac{\Psi}{T_r} \rfloor + 1}^{+\infty} p_k(n) = (1 - P_e)^2 \cdot$$
$$\cdot \sum_{n=\lfloor \frac{\Psi}{T_r} \rfloor}^{+\infty} \left[1 - (1 - P_e)^2\right]^n = \left[1 - (1 - P_e)^2\right]^{\lfloor \frac{\Psi}{T_r} \rfloor}. \tag{5}$$

Next, consider that a device generates an alarm message during the k–th update period. Then, at the $(k+1)$–th successful variable transmission production, the device signals the presence of acyclic data to the scheduler. The scheduler releases a delegated token and the device can transmit its alarm message. The last two operations are performed by means of acknowledged messages. Thus, we have the following decomposition for the alarm latency:

$$D = V + B_0 + B_1 + B_2, \tag{6}$$

where the first term, V, denotes the time between the alarm generation and the next time epoch that device will produce a variable (see Fig. 5). Then, since the time instants the device successfully transmits its variable act as regenerative points for that device, and since Poisson arrivals see time averages (the classical PASTA theorem, [18]), V can be modelled as the residual lifetime in a renewal process having renewal periods distributed as T_u. Then [19]:

$$v = \frac{t_u}{2E[T_u^2]}.$$

The second term, B_0, corresponds to the transmission time of the variable, and is distributed as T_{nack}. The other two terms correspond to the time spent for releasing the token (B_1) and successfully perform the acyclic activity (B_2). Since acknowledged services are used, both are distributed as T_{ack}. Then,

$$d = \frac{t_u}{2E[T_u^2]} + t_{nack} + 2t_{ack}. \tag{7}$$

The computation of t_{nack} and t_{ack} is deferred to the appendix.

In order to validate the analysis, numerical simulations have been performed by means of the ns2 tool [20]. In particular, 10^5 production requests were generated, and the corresponding variables produced upon correct message reception. The refresh period has been set to $T_r = 10$ ms. In Fig. 6 we depicted the behavior of the update period for $P_e = 10^{-2}$ and $P_e = 10^{-1}$. The small variations around the multiples of T_r are due to the randomness inherently present in the access mechanism. In particular values less than $10ms$ may be occasionally observed: this is due to the possible negative difference resulting from two subsequent backoff periods.

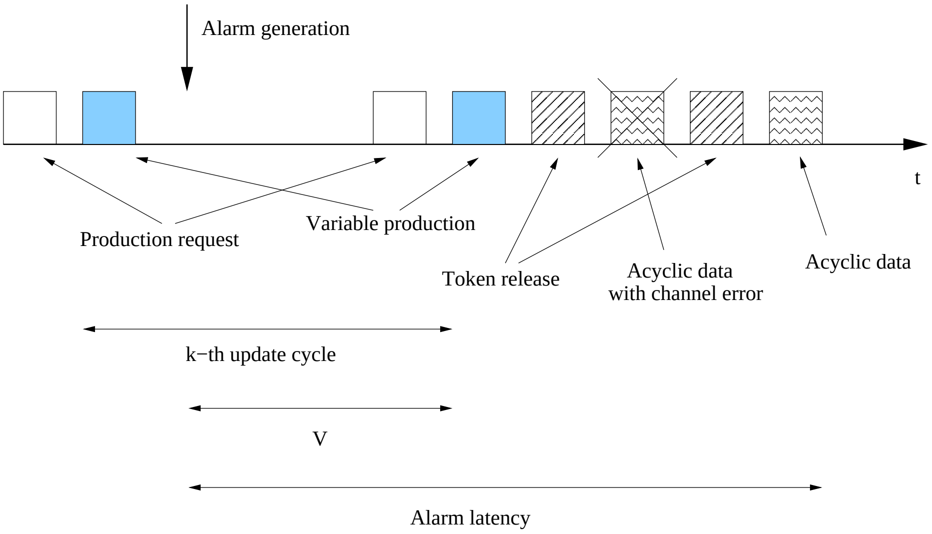
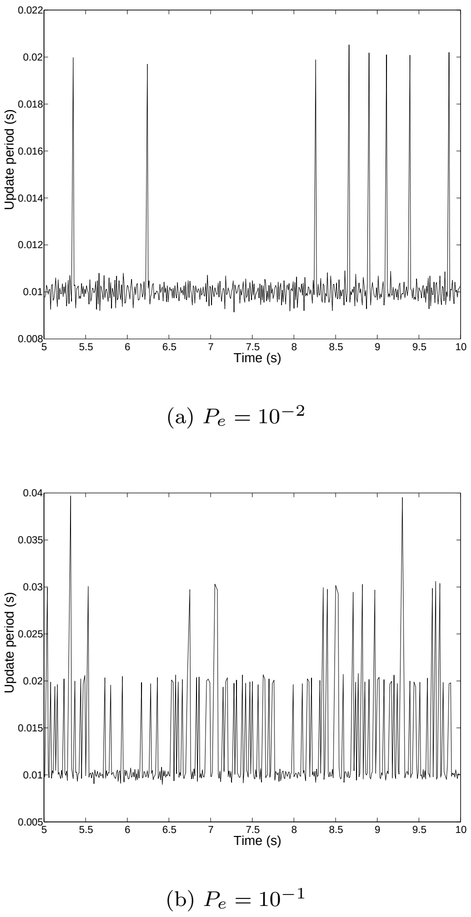


Figure 5. Decomposition of the alarm latency.



(a) $P_e = 10^{-2}$

(b) $P_e = 10^{-1}$

Figure 6. Update period behavior.

Some results for the mean update time are reported in Fig. 7, where simulation outcomes are compared with analytical results. As can be seen, for packet error probabilities, P_e less than 10^{-2}, the mean update time, practically, does not differ from the deterministic value (10 ms), whereas, for $P_e = 10^{-1}$ we have that t_u increases of about 25 percent. The update time jitter is plotted in Fig. 8 vs. the packet error probability. Also in this case we have that for P_e less than 10^{-2} the jitter may be neglected.

The outage probability is plotted, for $\Psi = 15, 20$ ms, in Fig. 9. On the whole, the good match between analytical and simulation data validates the proposed analysis.

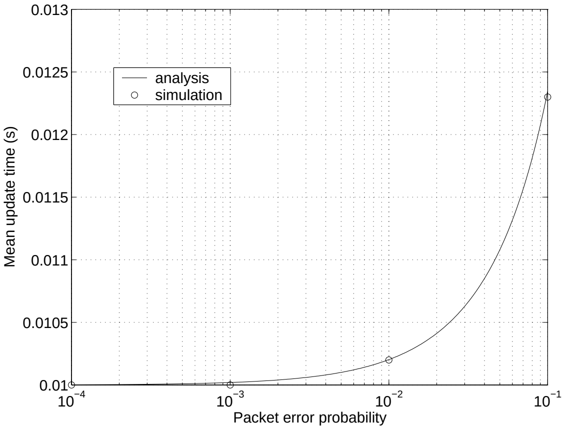


Figure 7. Mean update time vs. packet error probability, $T_r = 10$ ms.

As far as the mean alarm latency is concerned, some numerical results, for $T_r = 10$ ms, are depicted in Fig. 10, where the performance metric is evaluated for alarm packets of length $L = 30$ and $L = 100$ bytes. Once again, we may notice that for P_e less than 10^{-2}, the results obtained are encouraging, since alarm latency times are of the order of some milliseconds, which are values comparable with those of wired fieldbuses. Further, it is worth noticing that the per-

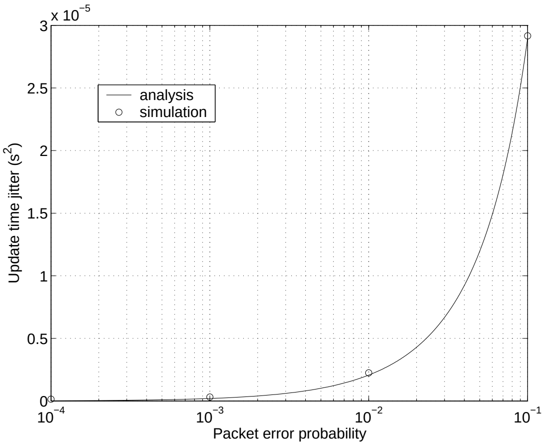


Figure 8. Update time jitter vs. packet error probability, $T_r = 10$ ms.

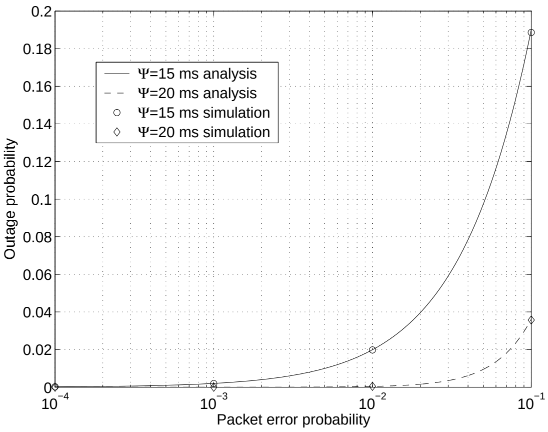


Figure 9. Outage probability vs. packet error probability, $T_r = 10$ ms.

formance, in terms of mean alarm latency, are almost insensitive to the packet size, a desirable feature from the point of view of system dimensioning.

5 Conclusions

In this paper we addressed the possibility of implementing a Producer–Consumer protocol on top of the IEEE802.11b WLAN. Since, similarly to the classical wired LANs, IEEE802.11 specifies the adoption of the 802.2 LLC as interface to the higher layers, we provided a mapping of the Producer–Consumer functions into the LLC services. In particular, unacknowledged services have been proposed for implementing the periodic exchange of variables, whereas confirmed services are employed to perform acyclic activities.

Under some simplifying but classical assumptions, an analytical framework has been presented in order to

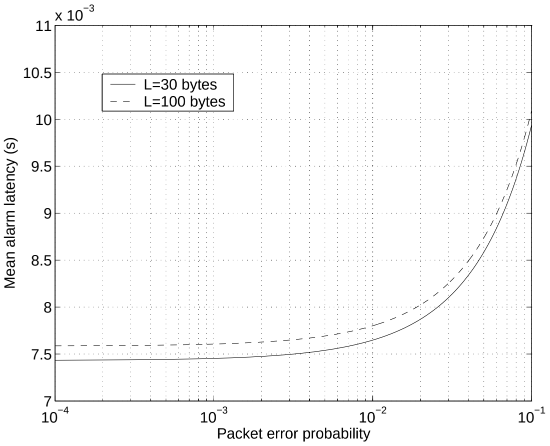


Figure 10. Mean alarm latency vs. packet error probability, $T_r = 10$ ms.

evaluate the network performance. In particular, we have focused on two metrics of interest, the update time jitter and the mean alarm latency. A performance index of interest for network dimensioning purposes, namely the outage probability, has also been derived. Results obtained from numerical simulations have been presented to validate the analysis.

On the whole, we may conclude that the proposed model presents performance figures which are not far from those of classical wired fieldbuses, and are suitable for most industrial applications.

Further developments of the presented work will consider both the impact of a more realistic channel characterization and an analysis of the network lifetime, where the presence of battery–limited devices is considered.

6 Acknowledgments

The work of D. Miorandi was partially supported by *Fond. A. Gini* and *CRBV*.

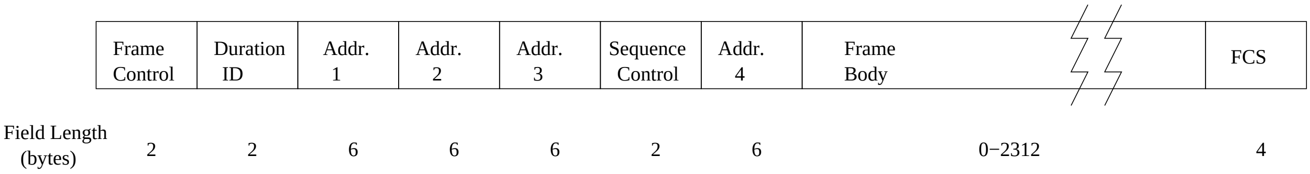
References

[1] R-fieldbus. [Online]. Available: http://www.rfieldbus.de

[2] *IEEE standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std., Aug 1999.

[3] A. Willig, M. Kubisch, C. Hoene, and A. Wolisz, “Measurements of a wireless link in an industrial environment using an IEEE 802.11-compliant physical layer,” *IEEE Trans. on Ind. Electr.*, vol. 49, no. 6, pp. 1265–1282, December 2002.

[4] A. Willig, “Polling-based MAC protocols for improving real-time performance in a wireless net-



	Frame Control	Duration ID	Addr. 1	Addr. 2	Addr. 3	Sequence Control	Addr. 4	Frame Body	FCS
Field Length (bytes)	2	2	6	6	6	2	6	0−2312	4

Figure 11. 802.11 MAC frame format.

Parameter	T_{slot}	T_{SIFS}	T_{DIFS}	T_P	T_{PHY}	CW_{min}
Value	$20\mu s$	$10\mu s$	$50\mu s$	$144\mu s$	$48\mu s$	32

Table 1. Parameters of 802.11b (long PLCP preamble).

work," *IEEE Trans. on Ind. Electr.*, vol. 50, no. 4, pp. 806–817, August 2003.

[5] *Profibus Standard: Translation of the German National Standard DIN 19245 parts 1 and 2*, Profibus Nutzerorganization e.V. Std., 1991.

[6] *Profibus DP Standard: Translation of the German National Standard DIN 19245 part 3*, Profibus Nutzerorganization e.V. Std., 1994.

[7] *RFC 791, Internet Protocol*, Defense Advanced Research Projects Agency, DARPA Std., January 1981.

[8] *RFC 793, Transmission Control Protocol*, Defense Advanced Research Projects Agency, DARPA Std., September 1981.

[9] *IEC 61158-3,4: Digital data communications for measurement and control - Fieldbus for use in industrial control systems - parts 3 and 4: Application Layer service definition and protocol specification, communication model type 8*, International Electrotechnical Commission Std., January 2000.

[10] *IEC 61158-3,4: Digital data communications for measurement and control - Fieldbus for use in industrial control systems - parts 3 and 4: Application Layer service definition and protocol specification, communication model type 3*, International Electrotechnical Commission Std., January 2000.

[11] *IEC 61158-3,4: Digital data communications for measurement and control - Fieldbus for use in industrial control systems - parts 3 and 4: Application Layer service definition and protocol specification, communication model type 1*, International Electrotechnical Commission Std., January 2000.

[12] *IEEE standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std., Aug 1999.

[13] *Supplement to 802.11-1999,Wireless LAN MAC and PHY specifications: Higher Speed Physical Layer (PHY) extension in the 2.4 GHz band*, IEEE Std., Sep 1999.

[14] *IEEE 802.2 Logical link control (with amendments 3, 6 and 7)*, IEEE Std., 1998.

[15] *"CANopen Application Layer and Communication Profile", CiA/DS301, Version 4.01"*, CAN In Automation, International Users and Manufacturers Group e.V. Std., June 2000.

[16] *"EN50325-2: Industrial communication subsystem based on ISO 11898 (CAN) for controller-device interface -Part 2: DeviceNet"*, European Committee for Electrotechnical Standardization, Std., June 2000.

[17] *"Road vehicles - Interchange of digital information - Controller area network for high-speed communication, ISO IS 11898"*, International Standard Organization, Std., November 1993.

[18] L. Kleinrock, *Queueing Systems.* New York: John Wiley & Sons, 1975.

[19] S. Karlin and H. M. Taylor, *A First Course in Stochastic Processes.* London: Academic Press, 1975.

[20] The network simulator - ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/

A Computation of the mean values of T_{ack} and T_{nack}

The 802.11 MAC layer, as described in §2, exploits a binary exponential backoff mechanism to control in a decentralized way access to the channel. Under the assumptions of §4, the exponential backoff will be used also to deal with channel errors. In particular, recall that, in the case of basic access (see assumption (i)), we have $m = m' = 4$. Further, the packet error probability is given by P_e. Disregarding the backoff, the transmission time of a 802.11 PDU, depicted in Fig. 11 is given by:

$$T_{data} = T_{DIFS} + T_P + T_{PHY} + \frac{L_{MAC} + L}{R_{data}} + T_{SIFS} + \\ + T_P + T_{PHY} + \frac{L_{ACK}}{R_{control}}, \quad (8)$$

where a MAC–layer ACK consists of $L_{ACK} = 112$ bits, the MAC–layer overhead is given by $L_{MAC} = 272$ bits, T_P and T_{PHY} represent the time necessary for the transmission of the PLCP preamble and of the PHY header, respectively (values are reported in Tab.1). We thus have:

$$T_{ack} = \sum_{n=1}^{k} \left(T_{data} + T_{bo}(n) \right), \quad (9)$$

where $T_{bo}(n)$ is the time spent in backoff at the n–th transmission attempt. Note that in (9) we optimistically assumed that a host is able to suddenly detect a packet loss; in reality it waits for a timeout to expire and then, since it did not get the ACK, double the value of CW and attempt transmitting again.

According to the protocol, $T_{bo}(n)$ (when expressed in slots) is uniformly distributed in the set $\{0, 1, \ldots, 2^{n-1}CW_{min} - 1\}$, so that $E[T_{bo}(n)] = T_{slot} \cdot \frac{2^{n-1}CW_{min}-1}{2}$. Note that the $T_{bo}(n)$ are independent, but not identically distributed, and that k, the number of transmission attempts, is a (truncated) geometrically distributed random variable of parameter $(1 - P_e)$. We start by computing:

$$\begin{aligned} E[T_{ack}|k] &= \sum_{n=1}^{k} \left(T_{data} + E[T_{bo}(n)] \right) = kT_{data} + \\ &+ T_{slot} \sum_{n=1}^{k} \frac{2^{n-1}CW_{min} - 1}{2} = \\ &= kT_{data} + \frac{T_{slot}}{2} \sum_{l=0}^{k-1} \left(2^l CW_{min} - 1 \right) = \\ &= k \left(T_{data} - \frac{T_{slot}}{2} \right) + \frac{CW_{min}T_{slot}}{2} \cdot \left(2^k - 1 \right). \end{aligned} \quad (10)$$

Then, applying the total probability theorem:

$$\begin{aligned} t_{ack} &= E[T_{ack}] = E\left[E[T_{ack}|k] \right] = \sum_{k=1}^{4} (1 - P_e) P_e^{k-1} \cdot \\ &\cdot \left[k \left(T_{data} - \frac{T_{slot}}{2} \right) + \frac{CW_{min}T_{slot}}{2} \cdot \left(2^k - 1 \right) \right] = \\ &= (1 - P_e) \left\{ \left(T_{data} - \frac{T_{slot}}{2} \right) \cdot \frac{1 + 4P_e^5 - 5P_e^4}{(1 - P_e)^2} + \right. \\ &\left. + CW_{min}T_{slot} \cdot \frac{1 - (2P_e)^4}{1 - 2P_e} - \frac{CW_{min}T_{slot}}{2} \cdot \frac{1 - P_e^4}{1 - P_e} \right\}. \end{aligned} \quad (11)$$

As far as the unacknowledged services are concerned, we easily recognize:

$$T_{nack} = T_{data} + T_{bo}(1). \quad (12)$$

Thus:

$$t_{nack} = T_{data} + T_{slot} \cdot \frac{CW_{min} - 1}{2}. \quad (13)$$