Analysis of Master–Slave Protocols for Real–Time Industrial Communications over IEEE802.11 WLANs

Daniele Miorandi and Stefano Vitturi
IEIIT-CNR, Department of Information Engineering, University of Padova
via Gradenigo 6/B, 35131 Padova (Italy)
Email: {daniele.miorandi, vitturi}@dei.unipd.it

Abstract— **The recent performance improvements of wireless communication systems are making possible the use of such networks for industrial applications, which typically impose severe requirements in term of both real–time communications and dependability. Several independent studies have highlighted that the IEEE802.11 Wireless LAN is one of the most suitable products for such applications. However, since such standard is only concerned with the lower layers of the communication stack, it is necessary to integrate it with appropriate protocols, typical of the industrial communications. In this direction, the protocols used by the traditional fieldbuses could represent an interesting opportunity. In this paper we consider one of these protocols, based on a Master–Slave architecture, and analyze the possibility of implementing it on top of IEEE802.11. After a description of how the Master–Slave functions could be mapped onto the IEEE802.11 services, we develop a theoretical model of the proposed communication architecture which allows for the evaluation of some performance metrics.**

I. Introduction

The use of communication networks in the industry is heavily influenced by the peculiar requirements of the application environment. In particular, if the network is employed at the lowest level of factory communication systems, also known as the "device level", where fieldbuses are traditionally used, it has to be able to cope with real–time communications between controllers and sensors/actuators. As an additional constraint, industrial plants are known to represent an hostile environment for communication systems, due to the presence of several kinds of electromagnetic noise. Moreover, it has to be considered that, often, the environmental conditions are characterized by the presence of dust, vibrations, and other adverse factors. In the past, the aforementioned considerations were sufficient to discourage the use of wireless systems for industrial applications. Recently, however, we have assisted to an impressive growth in the deployments of such networks in several fields of application, which, in turn, has led to increased dependability, performance improvement and cost reduction.

As a consequence, wireless networks are beginning to represent a viable choice also for industrial applications.

Thus, it is very likely that in the near future we will assist to a proliferation of hybrid wired/wireless implementations of factory communication systems. With such configurations it will be possible, for example, to use a wireless link to connect moving equipment which would otherwise be doomed to remain isolated.

Several researches have focused on the performance analysis of the most popular wireless networks in industrial environments. The most important of them is probably represented by the R-FIELDBUS project [1], whose architecture is described in [2], supported by the European Commission in the 5^{th} FP. Some interesting measurements on wireless networks for industrial applications are described in [3]; besides, hybrid wired/wireless systems have been considered in [4] and [5]. As a definite result of the above studies, the IEEE802.11 standard for wireless local area networking [6] has been indicated as a suitable candidate for industrial applications.

However, since the IEEE802.11 standard specifies only the lower layers of the communication stack, it is necessary to analyze how these may be integrated into an appropriate protocol stack for industrial communications. In particular, it seems a natural choice to place typical fieldbus upper layer protocols over the IEEE802.11 radio system. Most of such protocols can be divided into two main categories Master-Slave and Producer-Consumer.

In detail Master–Slave protocols make use of the lower layer services to implement a set of functions necessary for the correct data exchange among the devices connected to the network. Typical examples of such protocols are Profibus DP [7] and Interbus [8].

In this paper we consider the behavior of a Master–Slave protocol placed on top of the IEEE802.11 wireless LAN. More precisely, we propose a prototype of a Master–Slave protocol and discuss how it could be implemented using the communication services supplied by IEEE802.11. The protocol we propose does not require any change to the IEEE802.11 specification and consequently it may make use of commonly available network components, provided that they are suitable for industrial applications. An interesting possibility to this extent is given in [9]. Moreover, since IEEE802.11 networks may be easily integrated in internet/intranets, our proposal allows for the remote access of all the network components for configuration, diagnosis and maintenance purposes.

The rest of the paper is organized as follows: Section II illustrates the features of the IEEE802.11 standard. Section III describes the Master–Slave protocol prototype and discusses some protocol engineering issues. Section IV provides an analytical framework for performance evaluation of the proposed protocol. Section V concludes the paper.

II. Background: the IEEE 802.11 wireless LAN

IEEE802.11 [10] 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. We focus on the version currently deployed, known as 802.11b [11], which is able to provide data rates up to 11 Mb/s. 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. 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.

Finally, the IEEE802.11b standard 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.

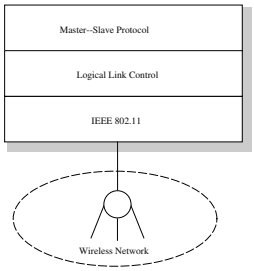

Fig. 1. Communication profile of the proposed Master–Slave protocol model

III. Implementation of the Master–Slave protocol over the IEEE802.11 wireless LAN

A. Generalities

Since the IEEE802.11, in accord with the LANs standardization, suggests the use of the Logical Link Control LLC [12] as interface towards the upper layers, we decided to adpot the same architecture. Thus, the communication profile of the proposed Master–Slave protocol is shown in Fig. 1.

LLC provides three types of services to the users, namely, connectionless acknowledged services, connection oriented unacknowledged services and connectionless unacknowledged services: they are supplied through Service Access Points (SAPs).

B. Protocol design

In Master/Slave protocols, one or more master devices have access to the transmission medium and poll the slaves which, conversely, may answer only when queried, so that they act as passive devices. Suitable techniques have to be used in order to grant to every master an access period sufficient to perform all the necessary operations on its slaves. In Profibus DP, for example, such a task is accomplished by means of a token passing scheme.

However, it has to be observed that most practical applications are based on networks comprising only one master device (monomaster). For such a reason, in this paper we will consider only monomaster networks.

The traffic generated on a network operating at the device level of factory communication systems is characterized by the exchange of two types of data: cyclic and acyclic. The cyclic traffic is due to the periodic exchange of both process states and commands which takes place between controllers and sensors/actuators. The acyclic data, on the other hand, usually correspond to unattended events such as, for example, alarms which, in the most serious cases may compromise the process operations, and which demand a prompt response in order to drive the system back to normal operation modes.

Hence, in order to obtain an efficient behavior, it is of crucial importance that cyclic data are updated with very low jitter and that little latencies are introduced in the transmission/acquisition of acyclic data. In these directions, some studies have appeared in the literature relevant to the most popular fieldbuses: among the most significant, Profibus has

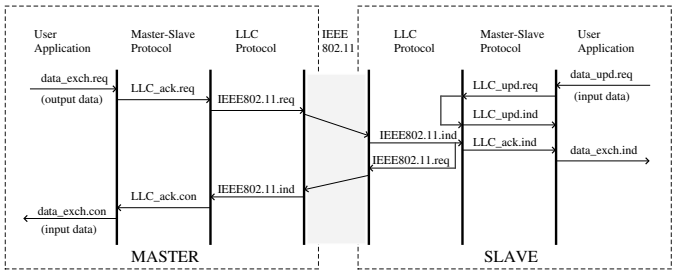

Fig. 2. Implementation of the cyclic data exchange

been deeply investigated in [13]; an analysis of WorldFIP is reported in [14], while in [15] the IEC fieldbus is considered. Indeed, the design of a Master–Slave protocol relies on a careful handling of both the aforementioned types of traffic. For example, in Profibus DP a Master executes the cyclic polling of the slaves which, when queried, may signal the presence of acyclic data (in this case they are referred as "diagnostic"). Then the master is forced to read the diagnostic at the end of the current polling cycle. Such a technique presents a drawback, in that the occurrence of acyclic data delays the beginning of the next polling cycle. Thus, if diagnostic requests are frequent, an undesired jitter affects the cycle time (i.e. the period with which the slaves are polled). Interbus [8] adopts a different technique: it assigns to each slave a predefined slot within a frame which is circulated repeatedly on the network. Since in such a slot, both types of traffic have to be dealt with, careful dimensioning is required. In this case, the jitter is eliminated, but a certain bandwidth waste is introduced. For the sake of completeness, it has to be remarked that a recent version of Profibus DP, named "synchronous" [16] is able to operate without jitter. In practice, the cycle time is maintained constant by adding a fixed interval at the end of each polling cycle. Such an interval has to be large enough to host, even in the worst case, all the acyclic activities.

The details of how cyclic and acyclic activities are implemented in the proposed Master–Slave protocol are given below.

1) Cyclic data exchange: The cyclic data exchange is realized by means of periodic queries sent by the master to the slaves. In order to implement such a function we propose to use the acknowledged connectionless services of LLC, as shown in Fig. 2. As it may be seen, the exchange starts with a request issued by the Master, carrying the output data. As a consequence, a request of an LLC acknowledged service is sent and the output data are encapsulated into an LLC frame passed to the IEEE802.11 MAC. The slave, which is notified of the data arrival by an indication of the LLC service, responds with a frame containing the input data which had been previously prepared by means of the data_upd service that in turn triggered the DL-REPLAY-UPDATE service of LLC, as shown by the primitives LLC_upd.req and LLC_upd.con.

2) Acyclic data transmission: Since this type of data may derive from critical events, it is of great importance that the data are correctly received. For this reason we chose to use, also in this case, the acknowledged connectionless services of LLC.

We propose three different techniques to handle acyclic traffic:

(i) The first technique, which for convenience will be referred to as *late* is analogous to that of Profibus DP and specifies that the master, at the end of the current polling cycle, queries the slaves that signalled in precedence the presence of acyclic data. This technique is based on the traditional assumption that slaves are passive devices which cannot access autonomously the network.

(ii) The second technique, named *current* allows a slave, when polled, to send directly acyclic data to the master. In practice the slave replaces the cyclic data with the acyclic ones. The master, when analyzing the received data, recognizes that they are acyclic (for example by means of a special code written in the LLC SAPs) and undertakes the appropriate actions. It is clear that, with this procedure, when a slave transmits acyclic information, the input data of that slave are not updated for one polling cycle. Nevertheless, such a solution is of ease implementation and has the advantage of reducing both the update jitter and the acyclic data latency.

(iii) The third technique, called *immediate*, exploits the decentralized nature of the IEEE802.11 MAC protocol and hence the capability of the devices to autonomously access the network. In this case, a slave station, when acyclic data are generated, immediately tries to send them to the master. Clearly, in this case collisions with the cyclic traffic may take place, and we rely on the ability of the MAC to resolve the contention for accessing the channel. It is worth mentioning that a protocol using this technique behaves slightly different from a traditional master-slave. However, we believe it is interesting to investigate such an additional possibility since it could enhance the protocol performances without increasing the implementation complexity.

IV. Performance Analysis

In this section we aim at evaluating the performance of the Master–Slave protocol when implemented over IEEE802.11. In particular, our analysis will focus on two metrics of interest: the update period jitter and the mean alarm latency.

The update period of a slave, T_u, is defined as the period with which the data to/from a slave are updated: since transmission errors may take place, 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 slave, of an alarm message and its successful transmission to the master. In order to keep the notation simple, we consider that all the slaves have the same features (symmetrical scenario); nevertheless, the framework naturally extends to asymmetrical networks.

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 [1];

(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.

[1]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.

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). 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. However, in case of bursty channels, the rate adaptation mechanism is likely to change the modulation scheme, in order to adapt to the channel state. 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. Further, let us denote by T_{ack} the transmission time of a packet sent with an acknowledged service and by N the total number of slaves. The derivation of the mean and variance of T_{ack} is lengthy and is omitted due to lack of space; the final expressions are reported in (12) and (13), while numerical values for T_{data} and the other system parameters can be found in [17].

A. *Late* technique

For the *late* technique, the update period T_u coincides with the cycle time T_c, defined as the time elapsed between two successive polls of the same slave for cyclic data exchange. We number the links from 1 to $2N$, in such a way that the $(2h-1)$–th link corresponds to the connection from the master to the h–th slave, and the $(2h)$–th one to the link from the h–th slave to the master. Let P_{alarm} denote the probability that any slave has an alarm message to send. Since the slave behaviors are independent, then the occurrence of M alarms in a cycle may be considered as M successes in N independent Bernoulli trials. Hence, M may be modelled as a binomial random variable with parameters (P_{alarm}, N). Assuming, for the sake of clarity and without loss of generality, that alarms are generated by the first M slaves, we can finally write the cycle time as:

$$T_c = \sum_{i=1}^{2N} B_i + \sum_{i=1}^{M} \left(B_{2i-1} + B_{2i}\right), \tag{1}$$

where B_i are independent identically distributed random variables which account for the times necessary to transmit the data. Since the LLC services we use are mapped onto IEEE802.11 acknowledged services we have: $B_i = T_{ack}$. In steady state, clearly, $P_{alarm} = 1 - e^{-\lambda t_c} \approx \lambda t_c$, so that some easy algebra leads to:

$$t_u = \frac{2Nt_{ack}}{1 - 2Nt_{ack}\lambda}. \tag{2}$$

The computation for the variance follows straightforwardly, and we obtain:

$$\sigma_{T_u}^2 = 2N(1 + P_{alarm})\sigma_{T_{ack}}^2 + 4NP_{alarm}(1 - P_{alarm})t_{ack}^2. \tag{3}$$

As far as the alarm latency is concerned, let us focus, without loss of generality, on an alarm generated by the first slave. Assume that the alarm is generated during the k–th cycle. Then, at the $(k+1)$–th cycle, the slave signals the presence of acyclic data to the master. After having performed all cyclic activities, the master polls the first slave asking for the transmission of the alarm message. Thus, we have the following decomposition for the alarm latency:

$$D = V + \sum_{i=2}^{2N} B_i + \left(B_1 + B_2\right), \tag{4}$$

where the first term, V, denotes the time between the alarm generation and the next time epoch slave 1 gains access to the channel. Then, since the time instants slave 1 starts performing its cyclic data exchange act as regenerative points for that slave, 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 from [19] we have: $v = \frac{E[T_u^2]}{2t_u}$. Hence,

$$d = \frac{E[T_u^2]}{2t_u} + (2N + 1)\, t_{ack}. \tag{5}$$

B. *Current* technique

In this case the cycle time is given by $T_c = \sum_{i=1}^{2N} B_i$, and is invariant with respect to the alarm generation statistics. The state of the k–th slave is updated when data messages are sent, which happens at every cycle with probability $1 - P_{alarm} = 1 - \lambda t_c$. Thus, we may write $T_u = \sum_{i=1}^{F} T_c(i)$, where $T_c(i)$ is the i–th cycle time duration and F is a geometric random variable, having probability mass function given by $P[F = k] = (1 - P_{alarm})P_{alarm}^{k-1}, \quad k = 1, 2, \ldots$. Since $\sigma_{T_c}^2 = 2N\sigma_{T_{ack}}^2$, the mean of the update is given by:

$$t_u = \frac{t_c}{1 - P_{alarm}} = \frac{2Nt_{ack}}{1 - 2Nt_{ack}\lambda}, \tag{6}$$

and the variance: $\sigma_{T_u}^2 = \frac{4N^2 t_{ack}^2 P_{alarm}}{(1-P_{alarm})^2} + \frac{2N\sigma_{T_{ack}}^2}{1-P_{alarm}}$, from which we have:

$$\sigma_{T_u}^2 = \frac{8N^3 t_{ack}^3 \lambda}{(1 - 2Nt_{ack}\lambda)^2} + \frac{2N\sigma_{T_{ack}}^2}{1 - 2Nt_{ack}\lambda}. \tag{7}$$

It is worth remarking that the *late* and *current* techniques present the same mean update period.

Similarly to the *late* algorithm, the alarm latency can be written as $D = V + B_1$, where, following a reasoning similar to that of the *late* case, we have $v = \frac{E[T_u^2]}{2t_u}$. Then,

$$d = \frac{E[T_u^2]}{2t_u} + t_{ack}. \tag{8}$$

C. *Immediate* technique

The *immediate* technique can be analyzed upon the assumption that the 802.11 MAC protocol is able to resolve all the contentions arising for the channel, thus avoiding collisions. Under such an assumption, the update period may be written as:

$$T_u = \sum_{i=1}^{2N} B_i + \sum_{i=1}^{H} B_i,$$

where H is a binomial random variable of parameters (P_{alarm}, N) which represents the number of alarms generated

by all the slaves during one cycle. Hence, the mean and variance of T_u turn out to be:

$$t_u = \frac{2Nt_{ack}}{1 - Nt_{ack}\lambda}, \tag{9}$$

$$\sigma^2_{T_u} = N(2 + P_{alarm})\sigma^2_{T_{ack}} + \\ + NP_{alarm}(1 - P_{alarm})t^2_{ack}. \tag{10}$$

Assuming that at one time at most one slave has alarms to send and that the access to the channel is fair (so that the master and the slave with acyclic data have the same probability of accessing the channel [2]), we have:

$$D = V + \sum_{i=1}^{R} B_i + B_1,$$

where V may be thought as the residual lifetime in a renewal process with renewal period equal to T_{ack} (since acyclic data requests may occur only when the master is polling the slaves and hence an acknowledged IEEE802.11 service is being executed) and R is a geometric random variable with parameter $\frac{1}{2}$. Hence:

$$d = \frac{7}{2}t_{ack} + \frac{\sigma^2_{T_{ack}}}{2t_{ack}}. \tag{11}$$

D. Performance comparison and numerical results

The performances offered by the three proposed techniques have been evaluated for various system parameters. For all scenarios of interest, the packet length is $L = 30$ bytes, whereas alarms are generated according to independent Poisson processes of intensity $\lambda = 0.1\ \ s^{-1}$, which corresponds to one request of acyclic data transmission every 10 s on average. In particular, in the first scenario we kept a fixed number of slaves, $N = 20$, and varied the packet error probability. Results for the mean update period are reported in Fig. 3. As expected, *immediate* algorithm slightly outperforms *current* and *late*. Some results for the update period variance are plotted in Fig. 4; as it can be seen, the *current* technique provides the best results in terms of update jitter. In Fig. 5 results for the mean alarm latency are reported; it can be seen that, for such metric of interest, the *immediate* technique clearly outperforms the other possibilities.

In the second scenario, on the other hand, we studied the scalability of the proposed communication protocol with respect to the number of slaves. Results for the mean update period, the update period variance and the mean alarm latency are reported, respectively, in Fig. 6, Fig. 7 and Fig. 8.

On the whole, we may conclude that the three techniques exhibit performance figures which are comparable with those of the wired fieldbuses, as described in [21] and [22]. Moreover, both the *current* and *immediate* techniques represent choices of interest for applications with specific requirements: the first for the case of very jitter–sensitive applications and the second for scenarios where it is required to respond as soon as possible to the arising of anomalous conditions.

[2]In reality, the 802.11 MAC provides only long–term fairness, and short–term unfairness may arise [20].

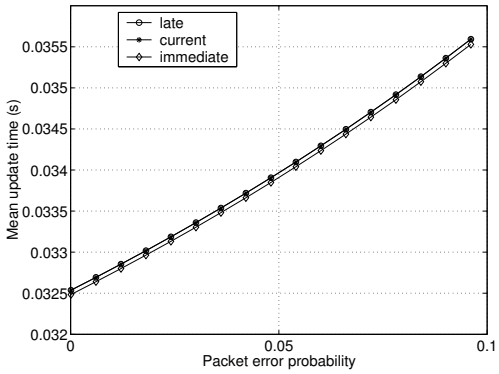

Fig. 3. Mean update period vs. packet error probability

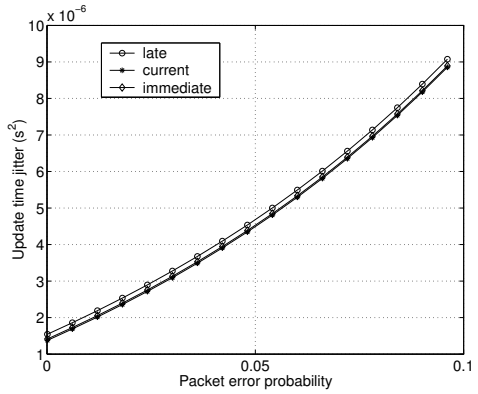

Fig. 4. Update period variance vs. packet error probability

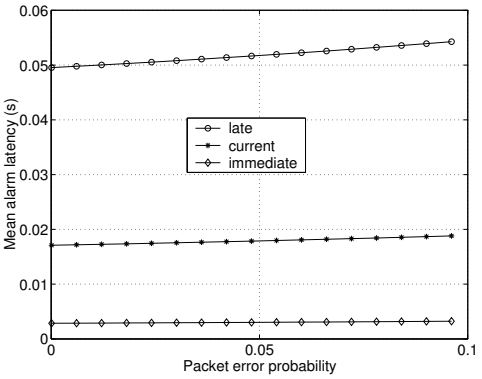

Fig. 5. Mean alarm latency vs. packet error probability

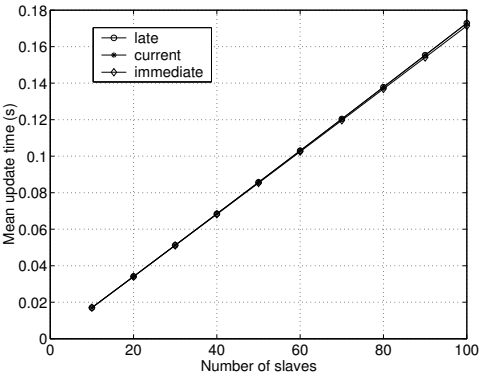

Fig. 6. Mean update period vs. number of slaves

V. CONCLUSION

In this paper we have analyzed the possibility of implementing a typical fieldbus protocol, the Master–Slave, on top of the

$$t_{ack} = \left(T_{data} - \frac{T_{slot}}{2}\right) \cdot (1 - 6P_e^5 + 5P_e^6) + (1 - P_e) CW_{min} T_{slot} \cdot \frac{1-(2P_e)^5}{1-2P_e} - \frac{CW_{min} T_{slot}}{2} \cdot (1 - P_e^5); \quad (12)$$

$$\sigma^2_{T_{ack}} = -t^2_{ack} + (1 - P_e) \sum_{k=1}^{5} E[T^2_{ack}|k] P_e^{k-1}; \quad (13)$$

$$E[T^2_{ack}|k] = k^2 T^2_{data} + 2T_{data} \cdot \sum_{n=1}^{k} E[T_{bo}(n)] + \sum_{n=1}^{k} E[T^2_{bo}(n)] + \sum_{n=1}^{k} E[T_{bo}(n)] \sum_{m \neq n} E[T_{bo}(m)]; \quad (14)$$

$$E[T_{bo}(n)] = T_{slot} \frac{2^{n-1} CW_{min} - 1}{2}; \quad (15)$$

$$E[T^2_{bo}(n)] = \frac{T^2_{slot} \left(2^{n-1} CW_{min} - 1\right)(2^n CW_{min} - 1)}{6}. \quad (16)$$

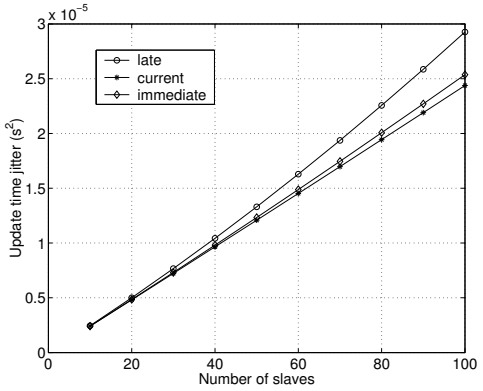

Fig. 7. Update period variance vs. number of slaves

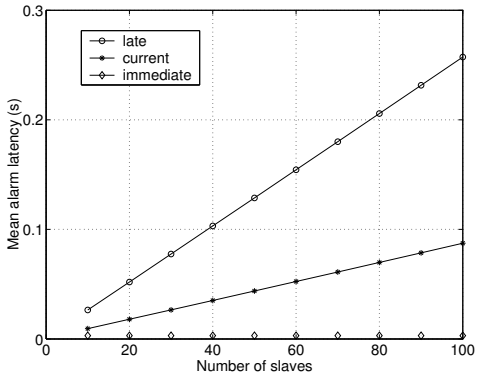

Fig. 8. Mean alarm latency vs. number of slaves

IEEE802.11 Wireless LAN. Following the structure proposed by such a standard for the interface to the upper layers of the communication stack, we decided to use the LLC services to map the Master–Slave functions on IEEE802.11. In particular, we have considered the acknowledged connectionless services as the most suitable to handle both the types of traffic we have analyzed: cyclic and acyclic.

The proposed model specifies a simple polling scheme for the exchange of cyclic data, whereas three different techniques for handling acyclic requests have been considered. In order to evaluate the performances of the protocol, we have developed a theoretical model which, under some simplifying but classical assumptions, allows us to investigate the behavior of the most important performance indexes.

REFERENCES

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

[2] L.Rauchhaupt, "System and device architecture of a radio based fieldbus–the rfieldbus system," in *Proc. of WFCS*, Vasteras, Sweden, 2002.

[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] S. Lee, K. C. Lee, M. H. Lee, and F. Harashima, "Integration of mobile vehicles for automated material handling using Profibus and IEEE 802.11 networks," *IEEE Trans. on Ind. Electr.*, vol. 49, no. 3, pp. 693–701, June 2002.

[5] L. Ferreira, M. Alves, and E. Tovar, "Hybrid wired/wireless Profibus networks supported by bridges/routers," in *Proc. of WFCS*, Vasteras, Sweden, 2002.

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

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

[8] *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.

[9] *Simatic Net Industrial Wireless LAN, White paper* 2003, Siemens AG. [Online]. Available: http://www4.ad.siemens.de

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

[11] *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.

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

[13] E. Tovar and F. Vasques, "Real–time fieldbus communications using Profibus networks," *IEEE Trans. on Industrial Electronics*, pp. 1241–1251, December 1999.

[14] L. Almeyda, E. Tovar, J. A. G. Fonseca, and F. Vasques, "Schedulability analysis of real–time traffic in WorldFIP networks: an integrated approach," *IEEE Trans. on Ind. Electr.*, pp. 1165–1173, October 2002.

[15] S. Cavalieri, A. D. Stefano, and O. Mirabella, "Optimization of acyclic bandwidth allocation exploiting the priority mechanism in the fieldbus data link layer," *IEEE Trans. on Ind. Electr.*, pp. 297–306, June 1993.

[16] *IEC61784 Final Draft International Standard: "Profile sets for continuous and discrete manufacturing relative to fieldbus use in industrial control systems"*, International Electrotechnical Commission Std., February 2002.

[17] D. Miorandi and S. Vitturi, "Performance analysis of producer/consumer protocols over IEEE802.11 wireless links," Department of Information Engineering, Univ. of Padova, Tech. Rep., 2004, submitted for publication. [Online]. Available: $www.dei.unipd.it/\sim jamaika/prod\_cons.pdf$

[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] M. Bottigliengo, C. Casetti, C. F. Chiasserini, and M. Meo, "Short-term fairness for TCP flows in 802.11b WLANs," in *Proc. of INFOCOM*, Hong Kong, 2004.

[21] G. Cena, L. Durante, and A. Valenzano, "Standard field bus networks for industrial applications," *Computer Standards and Interfaces*, vol. 17, no. 2, pp. 155–167, January 1995.

[22] G. Cena, C. Demartini, and A. Valenzano, "On the performances of two popular fieldbuses," in *Proc. of WFCS*, Barcelona, Spain, 1997.