Proceedings of the 2nd International Workshop on Real-Time LANs in the Internet Age (RTLIA), Porto, Portugal, July 1, 2003.

Wireless real-time communication protocol for cooperating mobile units[1]

Tullio Facchinetti[1], Giorgio Buttazzo[1], Marco Caccamo[2], Luis Almeida[3]

[1]DIS - University of Pavia
Pavia, Italy
tullio.facchinetti@unipv.it
giorgio@unipv.it

[2]University of Illinois at Urbana
Urbana-Champagne, Illinois, USA
mcaccamo@cs.uiuc.edu

[3]DET-IEETA, University of Aveiro
Aveiro, Portugal
lda@det.ua.pt

Abstract

Advances in wireless communication technology allow the development of distributed robotic applications consisting of teams of autonomous mobile units that can cooperate to achieve a common goal. Real-time communication among mobile wireless units, however, requires a careful management of the shared channel in order to achieve a certain level of predictability in message exchanging.

This paper proposes a new MAC level protocol that allows nodes to enter and leave the communication area, avoiding collisions due to simultaneous transmissions. The algorithm used for accessing the communication channel is based on EDF, which allows an optimal exploitation of the communication channel while guaranteeing timing constraints on real-time messages.

1. Introduction

Recent progress of wireless communication technology allows the development of distributed robotic applications consisting of teams of autonomous mobile units that cooperate to achieve a common goal. Future applications will require robots to autonomously operate in open environments for monitoring and exploration purposes. Sample applications include space missions, hazardous environment exploration, civil protection, demining and surveillance.

Typically, the communication system is based on a wired backbone, used to connect distant units. However, in many scenarios considered above a wired infrastructure cannot be guaranteed, hence a full autonomy of the robot team can only be achieved through an ad-hoc network. Since robots interact with the environment, most of the activities carried out by the team will be characterized by timing constraints that need to be enforced on the tasks to guarantee a minimum level of performance.

In wireless systems, real-time communication among cooperating robots requires a careful management of the shared channel in order to achieve a certain level of predictability in message exchanging. The primary objective is to reduce the communication overhead and avoid conflicts caused by simultaneous transmissions. In the absence of timing constraints, access conflicts can be resolved by suitable protocols that regulate message re-transmission. This solution, however, may introduce unbounded delays that may cause real-time tasks to miss their deadlines.

Some solutions for addressing this problem have been proposed in the real-time literature. In [2], the authors illustrate a real-time protocol at the MAC level based on the Earliest Deadline First (EDF) scheduling algorithm (firstly described in [5]) suitable for networks of steady sensors, but do not address the case of a node that wants to join or leave the team. Various deadline based scheduling policies such as Earliest Due-Date (EDD) [4], Delay-EDD [3], Jitter-EDD [11], Feasible-EDD [9] and Proactive-EDD [10] have been proposed for the real-time packet scheduling at the MAC level. They differ in how deadlines are calculated and assigned for each arrived packet, and how a packet is selected for service, but they do not permit any dynamic change in terms of nodes number. In [7], the BRAIN medium access protocol is investigated, a method for bandwidth exploitation and throughput maximization in a broadcasting environment. Although the approach allows an efficient exploitation of the available bandwidth, the periodic traffic is scheduled offline, thus no dynamic activities are handled. On the other hand, in [1] a centralized scheduling scheme is proposed to facilitate handling dynamic requests for periodic message streams, supporting on-line scheduling and admission control. However, it is centralized and based on a modified master-slave technique.

In this paper we propose a new protocol for the MAC

[1] This work has been partially supported by the European Union, under contract IST-2001-34820, and by the Italian Ministry of University Research (MIUR), under contract 2001097825_003.

level that operates directly over the wireless physical layer. This protocol allows nodes to enter and leave the communication area, avoiding collisions due to simultaneous transmissions. The algorithm used for accessing the communication channel is based on EDF. This allows an optimal exploitation of the communication channel while guaranteeing timing constraints on real-time messages. For the moment, we consider that all nodes are constantly connected, so we assume a scenario without hidden nodes. The detection of a new node in the communication area is carried out by listening to the channel during idle communication intervals.

2. Approach

The proposed method assumes that each node is synchronized with the others and knows the communication parameters of all the other nodes in the communication area. In this way, each node may construct the same EDF schedule for the entire team to know when it can transmit. Using this approach, transmission conflicts are avoided because each node sends messages at different times, according to the order dictated by the EDF scheduler.

Throughout the paper, it will be assumed that time is divided into slots of fixed length, and that nodes are synchronized on a slot basis.

Each periodic message is characterized by 5 parameters: the message identifier Id_i, the message size C_i, the period T_i, the relative deadline D_i, and the relative phase Φ_i, that is the time for the next instance with respect to a given temporal mark (the last 4 are expressed in number of slots). The set of all the Np periodic messages in the system is gathered in a communications table Γ defined as below, which is replicated in all active nodes.

$$\Gamma \equiv \{M_i\ (Id_i, C_i, T_i, D_i, \Phi_i),\ i = 1..Np\}$$

The message parameters determine the size of the table, which is an issue of great relevance when accounting for the communication overhead incurred by exchanging it among the nodes. Notice that this table must somehow be transmitted to new nodes joining the team so that they can construct the same schedule. According to the presented model, only a few bytes are required to encode the message parameters, e.g. one byte for each of the first two parameters and 2 bytes for each of the remaining four, amounting to 8 bytes per message. Just as an example, for a team of 10 robots, each broadcasting two periodic messages, the communication requirements table uses 160 bytes.

2.1 Joining the team

When a new node joins the team, the current communication requirements table must be updated in all participating nodes and must be sent to the joining one. The table transfer technique has also been used in [8] for the synchronization of backup masters in dynamic master-slave systems. In the remainder of this section we propose two different methods for the table exchange protocol, one based on a periodic table broadcast (PTB) and another based on a specific aperiodic request table (ART). In any case, the initial communication between the joining node and the current team is carried out using free slots.

PTB - Periodic Table Broadcast

- The bandwidth requirements of the entire team are periodically transmitted by the active units to allow each node to construct the EDF schedule. To minimize switching, such a message is transmitted by the current active node.
- When a new node wants to share the channel used by the team, it starts by listening to the channel to get the information on the communication requirements of the connected units.
- Using such data, the new unit verifies whether its bandwidth requirement can be satisfied based on the current load. Using EDF, this happens if the total load is less than one.
- If the new request can be accepted, the entering node reconstructs the current EDF schedule and will signal its presence and its communication requirements in the first available slot.
- All the nodes re-compute the schedule considering the new bandwidth requirements and the new schedule is started at a given time, so the new node is integrated in the team.

Figure 1 shows a sample sequence of the steps performed by a node that wants to join a team. At time t=9 a new node decides to join the team. It has to listen to the channel to wait for the scheduling table, periodically transmitted by the team.

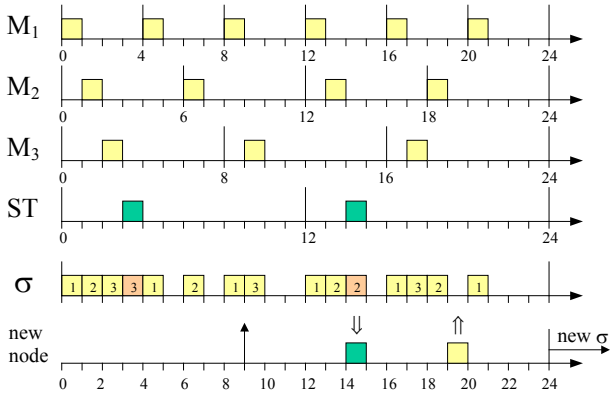

Figure 1: Acceptance of a new node according to the PTB method.

Notice that there must be a mechanism to select which robot from the team will actually send the table. Since there is no master or other node with special functionality within the protocol, a fully distributed

mechanism has been developed which reduces switching between transmission and reception. The idea is to select the node that just transmitted in the preceding slot to transmit the table message in the following slot(s). In the example in figure 1, the table is first transmitted by node 3 and then by node 2. If no node transmitted in the previous slot, then, the same node that transmitted the table in the last instance will transmit it again.

Once the table is transmitted, the new node can perform the acceptance test to verify whether it can be accepted in the team and, if so, it sends its request to join the team in the next idle slot after the time it completes the admission test. From this time on, all the nodes are aware of the new request and construct the new schedule, which is executed after a fixed number of slots after the request.

ART - Aperiodic Request Table

- The bandwidth requirements of the entire team are transmitted upon explicit request from a unit in a free time slot.
- When a new node wants to share the channel used by the team, it starts by listening to the channel to identify a free slot. The system must guarantee that the free time in any schedule is long enough to allow the new unit to transmit without conflicts.
- The new unit uses the free slot to send a request for receiving the communication requirements of the connected units.
- Using such data, the new unit verifies whether its bandwidth requirement can be satisfied based on the current load.
- If the new request can be accepted, the entering node reconstructs the current EDF schedule and will signal its presence and its communication requirements in the first available slot.

Figure 2 shows a sample sequence of the steps performed by a node that wants to join a team. At time t=1 a new node decides to join the team. It listens to the channel and sends its request in the first idle slot. At this point, all the nodes perform the admission test and, if the node can be accepted, the new scheduling table is transmitted by the last active node in the next available idle slot, after a certain time necessary to perform the admission test. Then, the joining node computes the new schedule, which is started after a fixed number of slots.

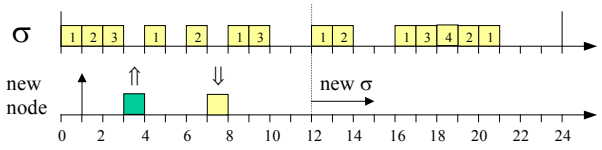

Figure 2: Acceptance of a new node according to the ART method.

All the nodes re-compute the schedule considering the new bandwidth requirements, so the new node is integrated in the team.

Notice that, when a node is broadcasting its data, possible errors occurring in the data packets or in the transmission channel do not affect the correctness of the schedule, which is generated only at precise time instants. The coherency and synchronization of the schedules can only be affected by inconsistent message delivery during the table update protocol when a node is joining or leaving. For the moment, we will consider atomic broadcast and thus, inconsistent messages will be addressed in future work.

2.2 Leaving the team

The case in which a robot wants to leave a team in the absence of hidden nodes can be exploited by two methods: (i) the leaving node advances an explicit request to all the other nodes, using a spare slot in the schedule; (ii) all the nodes listen to the channel and, when an idle slot is found in place of a node transmission, the node is automatically excluded from the schedule. This method however requires each node to send a message even when there are no data to be exchanged. Both methods require the nodes to agree on a predefined instant at which they re-construct the schedule without the leaving node.

3. Comparison

In a wireless environment, which is not as reliable as a wired network, it could be reasonable to have some sort of leadership to coordinate the incoming requests. Having a dynamic leader could make the algorithm more robust against either multiple requests or temporary network partitioning due to the presence of obstacles in the medium. Nevertheless, the approaches we presented above are not based on a master-slave communication paradigm, which, in such a robotics context, would have the disadvantage of being master dependent. In fact, a crash in the master would require a negotiation to appoint a new master node. The proposed approach, instead, is self-organizing, in the sense that there are no nodes with special functions in the team, and no communication overhead needs to be introduced in the protocol when a node crashes or leaves the team: all the nodes in the team notice a lack of transmitted packets from the dead node, so they simply start with a new schedule at a given time.

Comparing the two methods, we can see that:

1. The PTB method requires a periodic transmission of the communications table, whereas in ART it is transmitted upon request of the joining node. Hence, PTB introduces more overhead.
2. In the PTB method, the time to send the communications table can be pre-allocated in the schedule and taken into account as an additional periodic activity. With ART, it must be properly handled and guaranteed as a sporadic activity [6].
3. In both methods, the new node uses idle slots to

communicate with the other nodes in the team. As a consequence, any reclaiming algorithm (e.g., FRASH [2]) must be properly modified to avoid collisions when the new node starts using a free slot.

4. Using the PTB method, the new node receives the communications table from the team and may adapt its request to avoid an overload. Using the ART method, the active nodes receive the new request and, in case of overload, may decide to degrade the quality-of-service (QoS) they are receiving, by reducing the respective communication requirements according to a predefined QoS management policy. This way, enough bandwidth can be freed for the new node to be accepted. Further adaptation can still be done at the joining node side, reducing its own requirements.
5. In the PTB method, the overhead required to join a team is limited to a single free slot. After listening for the communication table (periodically transmitted), a new node should only signal its intention to join the team, transmitting its own communication requirements in the first following free slot. At a given time, all the nodes, including the new one, start with the new schedule. ART requires hand-shaking to complete the joining action because it is the joining node that requests the communication table. So, at least 2 slots are needed. Both methods require more than the least number of slots if a conflict occurs among two or more simultaneous joining nodes. The priority among the nodes could be established with classical methods.

4. Other issues and future work

One of the crucial aspects in the proposed protocol is the sustained synchronization of all schedulers, running in parallel in all nodes. The synchronization is based on the reception of packets. As soon as a packet is received, there is a slot timer in every node that is triggered. A distributed algorithm adjusts the slot timer count value in all nodes in order to maintain synchronization. During periods of communication inactivity, the timer continues to count and allows maintaining the nodes synchronized for a given time window that depends on the maximum clock drifts. To facilitate synchronization, all nodes are forced to transmit at least one message with a period not longer than that window, e.g. every 50 slots. Also notice that, between two consecutive slots there must be an inter-slot idle gap to accommodate clock drifts.

Another fundamental issue is the slot size, which impacts on the efficiency of bandwidth utilization. If too short, the overhead related to packet headers and tails will grow. If too long, its payload will not be effectively used by short messages and thus, part of the bandwidth will be wasted. A value of about 20 to 30 bytes of payload seems a good compromise to transport both environmental parameters as well as multimedia streams. Using a transmission rate of 1Mbit/s it is reasonable to consider a slot duration of about 250μs.

So far, we assumed that nodes communicate using a single channel. However, different robot teams can coexist in the same area using different frequency channels. Moreover, a robot located in an area common to two teams could communicate with both teams by switching frequency at proper times.

When considering power-aware issues, both methods require listening to the communication channel during idle slots in order to check the presence of nodes that want to join the team. A possible solution that could be applied to the proposed methods to save energy could be to listen to the beginning only, of idle slots. This would be enough to identify the presence of a joining node.

As future work, we will address the definition of the packet frame format as well as the issues of reliability in the presence of inconsistent message delivery and hidden nodes, which are particularly relevant in a wireless medium, typically characterized by its low reliability.

References

[1] L. Almeida, P. Pedreiras, J.A. Fonseca, "The FTT-CAN protocol: Why and How", *Trans. on Industrial Electronics*, 49(6), December 2002.

[2] M. Caccamo, L. Y. Zhang, L. Sha, and G. Buttazzo, "An Implicit Prioritized Access Protocol for Wireless Sensor Networks", *Proc. of IEEE RTSS'02*, Dec. 2002.

[3] D. Ferrari, and D. Verma, "A Scheme for Real-time Channel Establishment in Wide Area Networks", *IEEE Journal on Selected Areas in Communications*, Apr. 1990.

[4] J.R. Jackson, "Scheduling a production line to minimize maximum tardiness", *Management Science Research Project 43, University of California LA*, 1955.

[5] C.L. Liu, and J.W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment", *Journal of the ACM 20(1)*, pp. 40--61.

[6] Jeffay K. and D. L. Stone, "Accounting for Interrupt Handling Costs in Dynamic Priority Task Systems", *Proc. of IEEE RTSS'93*, pp. 212-221.

[7] L. Lo Bello, O. Mirabella et al., "An approach to comply with real-time constraints and bandwidth exploitation in distributed process control systems", *IEEE RTSS'99 - WiP*, Phoenix, AZ, USA, 1999.

[8] E. Martins, J. Ferreira, L. Almeida, P. Pedreiras, J.A: Fonseca, "An Approach to the Synchronization of Backup Masters in Dynamic Master-Slave Systems", *IEEE RTSS'02 – WiP*, Austin, TX, USA, 2002.

[9] S. Shakkottai and R. Srikant , "Scheduling real-time traffic with deadlines over a wireless channel", *Proc. of WoWMoM'99*, pp. 35-42, August 1999.

[10] K. Teh, P. Kong, and S. Jiang, "Proactive Earliest Due-Data Scheduling in Wireless Packet Networks", *Proc of ICCT'03*, April 2003.

[11] D. Verma, H. Zhang, and D. Ferrari, "Delay-jitter Control for Real-time Communication in a Packet Switching Network", *IEEE TRICOMM*, 1991.