An Approach to Fault Tolerant Clock Synchronization for Wireless Local Area Networks

Extended Abstract

Michael Mock[1], Reiner Frings[1], and Edgar Nett[2],

[1]GMD – German Research Center for Information Technology	[2]University of Magdeburg Institute for Distributed Systems
D-53754 St. Augustin	Universitaetsplatz 2
e-mail: {mock,frings}@gmd.de	D-39106 Magdeburg
Phone: +49 2241 142576	e-mail: nett@ivs.cs.uni-magdeburg.de
Fax: +49 2241 142324	Phone: +49 391 67 18346

1 Introduction

We are working on the development of a system architecture that enables autonomous mobile systems to cooperate in real-time using their own distributed infrastructure such as computing resources and wireless communication channels. Due to locomotion of the mobile systems, the cooperation is subject to strong real-time constraints. Furthermore, the systems share and compete for resources, e.g. network bandwidth. Our work aims at providing a hierarchy of real-time communication protocols that support the cooperation of autonomous intelligent systems. These protocols should support real-time and fault-tolerance properties. The development of the protocols is based on the IEEE 802.11 standard for wireless communication in local area networks [IEEE 97]. The protocols will be implemented in conformance with the infrastructure networks defined in the standard so that they can be used on standard hardware components (e.g. Lucent's WaveLan cards). The functionality of the standard is enhanced with respect to real-time and fault-tolerance properties either by adding additional protocol layers and, if necessary, by extending the standard implementation of the medium access control layer on the level of the device drivers.

On the lowest layer, which serves as basis for achieving real-time properties in the higher communication layers, a clock synchronization protocol implements a global time base for the cooperating systems. A global time base is the basic precondition for the real-time planning and execution of cooperative acts in such systems. The global time base can for instance be used to implement access control mechanisms in higher protocol layers based on the principle of time division multiplexing. Due to the unavoidable drift of local clocks, a global time base can only be achieved by the means of a clock synchronization protocol. In the context of mobile autonomous systems, the protocol must be based on wireless communication links. A global centralized protocol, such as provided by the GPS (Global Positioning System), does not solve the problem because the availability of the GPS is not guaranteed under all circumstances. Thus, a protocol working in a local group of cooperating systems is needed. The IEEE 802.11 standard includes a simple protocol for clock synchronization, which, however, only achieves limited precision and does not tolerate message losses. In this paper, we present a clock synchronization protocol that enhances the IEEE 802.11 standard by achieving high precision even in the presence of message losses.

2 Clock synchronization

In [Lam 78], the necessity of clock synchronization in general is motivated, [LuL 84] gives a lower bound on the precision that can be achieved (with deterministic algorithms) and proposes an algorithm achieving that precision. In [Cr 89], a probabilistic algorithm is introduced that achieves a higher precision (at the price of being non-deterministic). Examples of fault-tolerant algorithms for clock synchronization can be found in [LMS 85, KoO 87, LuL 88, Ver 92]. These protocols have in common that they spend a relatively high overhead for communication in order to achieve a sufficient level of precision and fault tolerance. Implicitly, they are designed for wired communication links that offer a sufficiently high bandwidth (e.g. Ehternet with 10 Mbits/sec) and a considerable reliability with respect to message delivery. In contrast to that, the wireless link has relatively low bandwidth (1 Mbit/sec, optionally 2 Mbits/sec) and a poor reliability. Therefore, a protocol for clock synchronization suitable to work in the application environment of mobile autonomous systems is required to achieve sufficient degrees of fault tolerance and precision with a low communication overhead. Furthermore, it should comply with the IEEE 802.11 standard, which is commonly accepted for wireless local area networks.

For the so-called infrastructure networks, the IEEE 802.11 standard already provides a master/slave clock synchronization mechanism. A special fixed node, the access point, is used as master. The access point coordinates the medium access for all stations that are reachable over the wireless medium. It determines alternating phases of medium access control: the "contention period" with CSMA based arbitration, and the "contention free period" with centralized medium arbitration. The access point initiates the contention free period by sending a high-priority message ("beacon frame"). This beacon frame includes a time-stamp that serves for synchronizing the local clocks of all slaves. The precision that can be achieved by this approach is bounded by the variance in the delay that is encountered between taking the time-stamp at the access point and receiving that time stamp at the slave stations.

3 The clock synchronization protocol

The basic idea of the proposed clock synchronization protocol is to increase the precision of the IEEE 802.11 clock synchronization protocol by exploiting the broadcast property of the wireless communication medium. As in [Ver 92, GS 94], we assume that message reception is tight, i.e., if any two receivers receive the same frame, they receive it approximately at the same time. Thus, we can make the precision achieved independent of the variance and delay of message delivery by the following procedure (that we have already applied in our clock synchronization protocol for the CAN-bus [GS 94]):

1. The master broadcasts an indication message. The transmission delay of this message is irrelevant for the precision of the clock synchronization, the protocol only assumes that message reception is tight.

2. Each slave (and the master) takes a local time-stamp right after reception of the indication message. We assume that the delay of an interrupt routine that time-stamps the reception of a frame is bounded and has a small variance.

3. The master sends its own time-stamp for the last indication message.

Now each slave can compare the master's time-stamp with its own time-stamp for the reception of the last indication message, compute the difference and adjust its local clock. The waste of bandwidth (two messages per synchronization) can be eliminated by an additional modification: the masters time-stamp for the last indication message now serves as new indication message, so we have only one synchronization message to be sent per synchronization period.

Let us now consider the fault-tolerance properties of the protocol in the context of the IEEE 802.11 standard. Applying the protocol in this context means that the access point acts as master, and that the beacon frame includes the time stamp of the reception of the previous beacon frame. We consider the access point to be stable because the IEEE 802.11 standard uses the access point as central coordinator during the contention free period. Since the master does not need to be aware of the existence of the slaves, failures and recovery of slave stations are transparent to the clock synchronization protocol. Thus, (fail-silent) site failures are tolerated by the protocol. However, the problem of message losses still has to be tackled. The number of message losses is considerable higher than for wired local area networks, because the wireless medium is unshielded and thus exposed to external interference. The protocol described so far achieves a high precision with a low communication overhead, but it does not tolerate message losses. A slave must receive two consecutive beacon frames in order to carry out synchronization. This is unacceptable when message losses occur frequently. For instance, loosing every second beacon frame would prevent the clocks from being synchronized at all.

Exploiting the tightness of message reception for synchronizing the clocks relies on analyzing the contents of the time-stamp message that refers to the reception of an earlier (indication) message. Thus, any fault-tolerant variation of the protocol still requires the successful reception of at least two different messages at the slave station. However, we can relax the requirement that these two messages must be consecutive. In order to tolerate up to (n-1) consecutive message losses, the time-stamp values for the last n synchronization messages are included in each synchronization message.

In more detail: Let sm_i denote the i-th synchronization message being sent by the master. Let tm_i denote the time-stamp of the master and cm_i the time-stamp of the slave for the reception of that message. Then, each synchronization message contains, besides its sequence number i, the values of tm_{i-n}, tm_{i-n+1}, tm_{i-n+2},…, and tm_{i-1}. Suppose that the last synchronization message received by the slave is sm_j at the slave's time cm_j. If the slave now receives sm_i, and if $i-j \leq n$, then the value of tm_j is included in sm_i and the slave can adjust its local clock based on the value of $cm_j - tm_j$.

Thus, on the reception of a synchronization message, a client can synchronize its local clock with the master clock if it has received at least one of the preceding n synchronization messages. An analysis of the precision of the protocol as well as its implementation on Windows NT network drivers for Lucent WaveLan PCMCIA cards are currently under way.

4 References

[Chr 89] F. Cristian. *Probabilistic clock synchronization*. Distributed Computing 3, 1989, pp. 146-158.

[GS 94] M. Gergeleit, and H. Streich. Implementing a Distributed High-Resolution Real-Time Clock using the CAN-Bus. 1st international CAN-Conference 94, Mainz, 13.-14.09., CAN in Automation e.V., Erlangen 1994.

[IEEE 97] IEEE. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications. IEEE Std. 802.11-1997, November 1997.

[KoO 87] H. Kopetz, and W. Ochsenreiter, Clock Synchronization in Distributed Real-Time Systems. IEEE Transactions on Computers, Vol. C-36, No. 8, August 1987, pp. 933-940.

[Lam 78] L. Lamport. Time, clocks, and the ordering of events in a distributed system. Commun. Ass. Comput. Mach., Vol. 21, July 1978, pp. 558-565.

[LMS 85] L. Lamport, and P. Melliar-Smith. Synchronizing Clocks in the Presence of Faults. Journal of the ACM , 32(1):pp. 52-78, January 1985.

[LuL 84] J. Lundelius, and N. Lynch. An upper and lower bound for clock synchronization. Inf. Control 62, 1984, pp. 190-204.

[LuL 88] J. Lundelius, and N. Lynch. A New Fault-Tolerant Algorithm for Clock Synchronization. Inf. Computing 77, 1988, pp.1-36.

[VeR 92] P. Veríssimo, and L. Rodrigues. A posteriori Agreement for Fault-tolerant Clock Synchronization on Broadcast Networks. 22th Int. Symp. on Fault-Tolerant Computing, Boston, July 1992.