Improving Real-Time Characteristics of a COTS Local Area Network *

Carlos Almeida
DEEC, Instituto Superior Técnico
Universidade Técnica de Lisboa
Av. Rovisco Pais - 1049-001 Lisboa, Portugal

José Rufino
DEEC, Instituto Superior Técnico
Universidade Técnica de Lisboa
Av. Rovisco Pais - 1049-001 Lisboa, Portugal

Abstract

The widespread use of computers and communication networks creates a distributed computing environment where there is the demand to build applications with every increasing requirements in what concerns dependability and real-time characteristics. However, this situation induces requirements on the communication subsystems that cannot be easily satisfied by most existing infrastructures without adaptations. These systems are not usually fully synchronous, presenting a high variability in response time that makes it difficult to achieve real-time operation. To improve this situation, we propose the use of the quasi-synchronous approach, where we use a small synchronous part of the system to control and validate the other parts. In the context of this paper, our target communication infrastructure is the ISO 8802/3 LAN (Ethernet), which is a low cost network with a large installed base. Being able to improve the real-time characteristics of such environment offers a very cost-effective solution to a large class of applications. In order to be able to build our proposed architecture in this setting, we introduce some mechanisms to enforce the real-time characteristics of the access control layer.

Keywords: real-time communications, quasi-synchronous systems, fault-tolerance, medium access control, COTS local area networks

*This work was partially supported by FCT, through Project PRAXIS/P/EEI/14187/1998 (DEAR-COTS), and by FEDER and national funds through POSI program.

1 Introduction and motivation

In the last decade there has been a tremendous increase in the use of computers and communication networks. This creates a distributed environment where new applications can be, and are, built everyday. These new applications have increasing requirements in what concerns dependability and real-time characteristics. Even existing applications that were traditionally done in a more centralized way, want to take advantage of these new distributed computing environments. This situation induces requirements on the communication subsystems that cannot be easily satisfied by most existing infrastructures without adaptations.

Although some specific applications have strict requirements (hard real-time) that imply the use of dedicated solutions, there are other applications with less strict requirements, that, although having some real-time and fault-tolerance requirements, want nevertheless to use generic components due to functional and/or cost restrictions. In some cases there is already a large installed base that is not easy (or it is too expensive) to substitute. In this situation dedicated solutions are

not cost-effective.

Despite the problems related to fault-tolerance and real-time have been addressed, and are reasonable well understood, in the context of synchronous systems, they are much more difficult to solve (if even possible) in systems that are not fully synchronous, which is the case of most general-purpose distributed environments. In order to provide support for the development of these new classes of applications, and there is a demand for it, those issues (dependability and real-time) must be addressed in this context.

A set of communication network technologies (e.g. ATM [5]) have improved the synchronism properties of distributed environments. However, they are not always fully synchronous. They are at most *quasi-synchronous* [14]. Only a small part of the system can be considered as synchronous. The rest has a more dynamic behavior exhibiting, for a given activity, worst-case delays that are much higher than the normal delays[1] (see Figure 1).

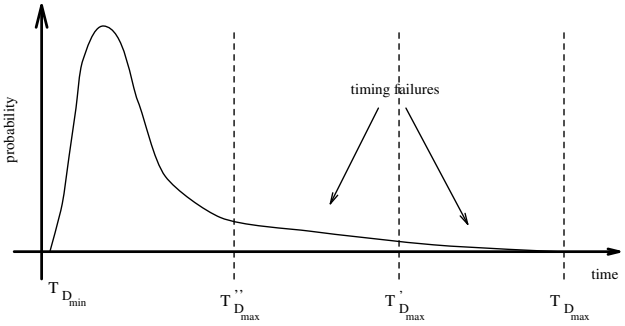

Figure 1: Distribution function for message delivery time (T_D) in a *quasi-synchronous* system. The worst-case is much higher than the normal case. Assuming as maximum a value closer to the normal case (due to practical reasons) increases the probability of having timing failures.

Real-time applications developed for this type of environment face the following dilemma: if they use always the worst-case times, they may become useless because those times are far away from the normal times; however, not using them may lead to timing failures, which, if not correctly handled, may lead to system failure.

Although it may not be possible to provide full timeliness (real-time guarantees to all activities) in this type of environment, it is desirable and important to provide some form of control to detect timing failures and provide the mechanisms to ensure safety in a timely fashion. Based on those mechanisms, applications can reach a safe state before stopping, or switch in a controlled manner between several different qualities-of-service, depending on global system evolution.

We have been addressing these problems (related to partial synchrony) both at model level and system level. It is our goal to address all the aspects required to build an usable system. This implies both the aspects related to processing and communications. Our approach (*the Quasi-Synchronous approach*) is to use a small synchronous part of the system to build components able to control and validate the other parts of the system.

In the context of this paper we are more concerned with the communication subsystem. We want to address the feasibility and usefulness of improving the real-time characteristics of a COTS (Commercial Off-The-Shelf) communication infrastructure in order to support new classes of real-time applications in a cost-effective manner. The target communication infrastructure is the ISO 8802/3 LAN (Local Area Network), commonly known as Ethernet.

There are many generic applications built on top of this low-cost communication infrastructure, and the desire to build applications with more demanding real-time and dependability requirements. One possible application would be in the area of shop floor (or in the area of an improved desktop, provided that there are some restrictions on configurations changes). However, Ethernet lacks determinism in the medium access control (MAC), which restricts the type of applications that can be built directly on top of this network. To solve

[1] These delays can be related to execution if it is a CPU related activity, or transmission in the case of a network related activity.

this problem, the MAC has to be enhanced by enforcing real-time characteristics. In order to preserve maximum compatibility (and thus avoid high costs in new equipment), this has to be done with minimal impact on the existing infrastructures.

The paper is organized as follows: in the next section the main problems to solve are identified. In Section 3 we describe our quasi-synchronous approach and in Section 4 we explain how to deal with medium access control. In Section 5 some previous and related work is presented. The paper ends with the conclusions and some considerations about future work.

2 Main problems to solve

As represented in Figure 1, in the type of environment that we are considering the synchronism is not perfect. The best that is possible to achieve is to have a small part of the system that can be considered as synchronous, but the other parts exhibit a more dynamic behavior where it is possible to have a high variability in response time.

There is indeed some uncertainty in communication delays in scenarios where load is not completely controlled. Due to dynamic characteristics of applications and/or environments, it is not usually cost-effective to have a resource adequacy policy that would prevent overload scenarios. This way we have to deal with the possibility of having some timing failures.

Although this situation makes this type of environment not suitable for safety-critical hard real-time applications (that would imply the use of a resource adequacy policy), there are other real-time applications, with less strict timeliness requirements, that can still be built in such scenarios and offer an acceptable service, provided that there are validation mechanisms.

The main problem to solve is that just following a best-effort policy is not adequate for most real-time applications. Most of them are not intrinsically tolerant of timing faults.

The high variability in response time that is present in this type of environments makes it difficult to build real-time applications that are both efficient (from the point of view of timeliness) and safe. There is a tradeoff between tight timeliness and timing failures. If, due to practical reasons, a "worst-case" is assumed that is smaller than the real worst-case, then the probability of having timing failures increases, as shown in Figure 1.

To deal with these timing failures, it is necessary, at least, to have mechanisms to detect those situations in a bounded time. This is needed in order to achieve safety in a timely fashion. Applications may need to stop in a safe state or dynamically adapt, in a controlled and timely way, the offered quality-of-service. In order to do this, we propose the use of the quasi-synchronous approach.

In this approach, that is explained in the next section, a small synchronous part of the system is used to control and validate the other more generic activities that have more relaxed synchronism properties. In order to be able to build the synchronous channel in our target communication infrastructure, we also have to deal with, and improve, the access control mechanisms to the communication medium.

3 The quasi-synchronous approach

A synchronous system is one that exhibits known bounds on process execution times, message transmission delay and local clock rate drift. An asynchronous system is one where there are no such bounds. However, practical scenarios are usually not so well defined. Most systems, although not being fully synchronous, are not completely asynchronous either. They exhibit some form of synchronism. In this category are what can be called *quasi-synchronous* systems [14].

A *quasi-synchronous* system can be modeled as if it was a synchronous system, in the sense that there are bounds on process execution times, message transmission delay and

local clock rate drift, but some or all of those bounds are not precisely known, or have values that are too far from the normal case, that in practice one must use other values (closer to the normal case). In both cases it means that there is a non-null probability that the values we pick are not correct. This is a realistic scenario when there are situations of overload. Tight synchronism properties are restricted to a small part of the system: a few high priority activities, and a small bandwidth channel for high priority messages.

In the quasi-synchronous approach, this small synchronous part of the system is used to build components able to control and validate the other parts of the system, thus making it possible to achieve safety in a timely fashion. This approach does not solve all timeliness problems *per se*, but can be used by applications to reach a safe state before stopping, or switch in a controlled manner between several different qualities of service. Together with group communication protocols and a hierarchy of group management it can also provide an efficient support for the use of active replication.

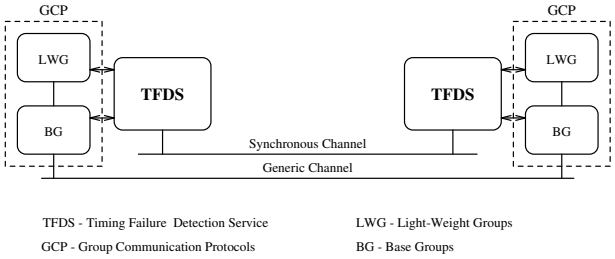

Figure 2: The quasi-synchronous approach architecture

The proposed architecture is represented in Figure 2. A timing failure detection service (TFDS), supported on a small bandwidth synchronous channel, is used to disseminate, with timeliness guarantees, control information. That control information is used by the communication protocols to validate their properties in a safe and timely manner [2].

The implementation of TFDS may require the use of a specific network or a dedicated channel in a more generic network. We can exploit several approaches. Depending on what is available and on the existing resources for a given application, one may choose the approach that best fits in a given scenario. If an ATM network is available, for example, TFDS can be implemented using a channel with better guarantees than the channels used for normal communication. In other settings, one may need to reserve a small part of the global resources in order to be able to build a correct TFDS. In the context of the work presented in this paper, that channel is obtained by using medium access control mechanisms, and by reserving a small bandwidth for that propose. Generic data may or may not have enough bandwidth at a given moment (situations of overload), but the control information associated with TFDS will have the required bandwidth.

4 Approaches to enhance medium access control

Our target communication infrastructure is the ISO 8802/3 LAN (Local Area Network), commonly known as Ethernet. However, Ethernet lacks determinism in the medium access control (MAC), which restricts the type of applications that can be built directly on top of this network. To solve this problem, the MAC has to be enhanced by enforcing real-time characteristics.

The fundamental obstacle to the utilization of the ISO 8802/3 LAN technology in real-time applications concerns the non-determinism of the exponential backoff algorithm used in the resolution of collisions. Nevertheless, with the appropriate design options, that scenario can be modified.

In order to preserve maximum compatibility (and thus avoid high costs in new equipment), this has to be done with minimal impact on the existing infrastructures.

One way of achieving this goal (for applications with coarse timeliness), is to build a layer on top of the exposed 8802/3 MAC interface, to

control application access to the network and prevent unauthorized direct access. This can be done using approaches such as token-based or TDMA (Time Division Multiple Access).

Another method of securing real-time characteristics can be built as an extension to the standard ISO 8802/3. Standard network attachment controllers support a comprehensive set of functions concerning the control of the access to the shared communication medium. Those functions include: buffer management, frame encapsulation, validation of frame correctness and overall medium access control. In the standard ISO 8802/3 LAN [6], collisions in the access to the broadcast medium are resolved through a non-deterministic exponential backoff algorithm, that constitutes the fundamental impairment to obtain a real-time behavior out of this LAN infrastructure.

One possible solution for a real-time variant of the ISO 8802/3 LAN requires the modification of the built-in collision resolution algorithm and the implementation of all the medium access control functionality in an application specific integrated circuit. However, such a solution requires a long design cycle and exhibits a high development cost. Furthermore, it does not allow the utilization of off-the-shelf "legacy" network interface cards.

A more simple solution does not exhibit such shortcomings. It requires only the modification of the external behavior of the collision resolution algorithm and it can be implemented through an interceptor to the standard PHY-MAC interface, to be built as an extension to the standard LAN infrastructure. The operation of the standard MAC protocol is prevented to proceed when a collision occurs, through the manipulation of the relevant signals at the PHY-MAC interface, by the ISO 8802/3 MAC interceptor. A deterministic resolution of collisions is then carried out by the MAC interceptor, through a simplified algorithm that can either be based on fixed identifiers or on the use of a dynamic message identifier captured from the frame header. The standard MAC sub-layer resumes normal operation at the end of this contention process.

The ISO 8802/3 MAC interceptor can be designed as an extension to the standard and can be easily implemented as a plug-in adapter to existing off-the-shelf network interface cards (NICs). Attachment to classical 10BASE-5 and modern 10BASE-T physical interfaces are possible.

4.1 Handling collisions and avoiding collisions

The way collisions are handled in ISO 8802/3 (CSMA/CD Ethernet) implies the possibility of having unbounded worst-case access times. This is a major problem if one wants to support real-time applications.

Keeping the native Ethernet MAC at low level, implies that, without other measures, any collisions can lead to potentially large resolution times. Avoiding collisions it is, thus, of utmost importance. That is the goal of the high-level approaches described above – to have access control algorithms that serialize the access to the medium, thus avoiding collisions. In normal operation there will be no collisions if all nodes follow those access control algorithms. Collisions will be restricted to the initialization phase, and to situations of possible failures.

To further minimize the impact of those collision scenarios, it is important to bound the resolution time, and to reduce that resolution time as much as possible. In order to achieve that, there are several aspects that can be addressed.

Ethernet controllers are typically configured to automatically retry a transmission when a collision occurs. The maximum number of transmissions upon consecutive collisions is sixteen. Only after that, an unsuccessful transmission is aborted and notified to upper layers. Due to the exponential backoff algorithm, this period of time can be significantly large. If new nodes are allowed to enter the contention, the collision resolution time may become unbounded. This situation must be avoided.

Some controllers allow a configuration where they do not automatically retransmit after a

collision, making instead the notification to the upper layers immediately after the first collision. Although this requires retransmission handling at upper layers, it allows a tight control over the collision resolution time. Unfortunately, not all controllers support this functionality.

In order to avoid a possible unbounded collision resolution time, it is important to make sure that there are no new nodes trying to access the network when there is already a collision resolution in progress. This can be done at upper layers by not sending messages to the controller in such a scenario. The sending is postponed until all messages already in the controllers are successful transmitted, or until they are dropped due to excessive collisions. This way, although the collision resolution time can still be potentially large, it will be bounded.

At upper layers the access control can be based on token, TDMA, or a mix of both. In any case the main idea is to avoid collisions. Once control is gained, and during normal operation, no collisions occur. However, there are special situations where additional care must be taken. That is the case of initialization, and handling of token loss, for example.

Although initialization is a specific case where additional delays are usually tolerable, and sometimes unavoidable, it is important to reduce those delays as much as possible. In the other scenarios (interferences during normal operation) minimizing those delays is even more important.

In order to do that, one may choose to start with a well-known network configuration (statically defined). All modifications are done in a controlled way, and current configuration information is kept in a consistent state in a distributed fashion. The goal of this approach is to avoid long worst-case scenarios.

5 Previous and related work

The present work is supported on previous work done in the context of the quasi-synchronous model. We have been addressing these problems of having real-time applications in systems that are not fully synchronous. Besides presenting the model [14, 4], we have proposed group communication protocols that provide message early-delivery [1], and we explained the use of the timing failure detection service and its architecture [2]. In [3] we describe the protocols associated with the management of a hierarchy of groups, and explain how we can handle timing failures in a quasi-synchronous system by using light-weight groups. As we show in that paper, the quasi-synchronous approach can be very useful in the handling of timing failures in an active replication scenario, provided that an independent failure mode can be assumed. Although the assumption of an independent failure mode is not always realistic, there are many situations where it can be considered.

In what concerns mechanisms to handle medium access control, the use of a token approach was also followed in the Rether project [12], that aims to achieve network bandwidth guarantees required by multimedia applications. An off-the-shelf Ethernet infrastructure without any hardware changes was used. A token passing scheme regulates the access to the Ethernet segment.

The other flavor of medium access control aims to address applications with "tight" real-time guarantees that cannot be satisfied in general by high-level access methods. Despite the existing research with regard the definition of deterministic variants of CSMA (Carrier Sense Multi Access) methods [7, 8], its use is not widespread. One reason could be the lack of attachment controllers supporting that functionality. Popular implementations of inexpensive network interface cards (NICs) use the non-deterministic variant.

One possible solution for the provision of real-time guarantees in CSMA networks would require the implementation of those deterministic algorithms in application specific integrated circuits, that should also include the remaining functionality of network attachment controllers. However, such a solution is not

cost-effective, due to its long design cycle and high development cost. Furthermore, it makes no use of existing off-the-shelf "legacy" equipment.

A less expensive similar alternative previously applied with success to the ISO 8802/4 LAN (Token-Bus) is: use a structured approach to the problem [13]; fully characterize the network timing behavior [9]; define solutions to improve that behavior, either through network planning and parameterizing [10] or by using special-purpose MAC interceptors [11].

6 Conclusions and future work

In this paper we addressed the problem of improving the real-time characteristics of a COTS local area network. Our target communication infrastructure is the ISO 8802/3 LAN, commonly known as Ethernet. There is a large installed base, and many generic applications built on top of this low-cost communication infrastructure. There is also the desire to build new applications with more demanding real-time requirements. However, this situation induces requirements on the communication subsystem that cannot be easily satisfied without adaptations. Ethernet lacks determinism in the medium access control (MAC), which restricts the type of applications that can be built directly on top of this network. The high variability in response time that is present in this type of environment makes it difficult to build real-time applications that are both efficient (from the point of view of timeliness) and safe. In order to deal with the possibility of having timing failures, it is not enough to just follow a best-effort policy. We need mechanisms to detect and handle those situations in a bounded time, so as to preserve safety. To do this, we propose the use of the quasi-synchronous approach.

In the quasi-synchronous approach, a small synchronous part of the system is used to build components able to control and validate the other parts of the system (with more relaxed synchronism properties), thus making it possible to achieve safety in a timely fashion. This approach does not solve all timeliness problems *per se*, but can be used by applications to reach a safe state before stopping, or dynamically adapt, in a controlled and timely way, the offered quality-of-service.

In order to be able to build the synchronous channel in our target communication infrastructure, we have to deal with, and improve, the access control mechanisms to the communication medium. To solve this problem, the MAC has to be enhanced by enforcing real-time characteristics. In order to preserve maximum compatibility (and thus avoid high costs in new equipment), this has to be done with minimal impact on the existing infrastructures.

By restricting synchronism properties to a few system modules, which makes its implementation viable in a larger setting, it is possible to build an infrastructure that supports the development of fault tolerant real-time applications in scenarios where is was not possible before.

It is not our goal to support the development of safety-critical hard real-time applications on environments that are not fully controlled. It is our goal to use an approach that allows the improvement of real-time characteristics, and at same time improve the validation of situations where timeliness goals are not fully achieved.

We use a best-effort policy to try to fulfill application timeliness requirements, but we also validate safety properties. If, by the assumed deadline, it is not possible to obtain the desired results, one can opt to stop in a fail-safe state, or one can relax timeliness requirements and wait for late results.

The characteristics of the validation mechanism that we refer above (TFDS) puts some restrictions on the type of environment where it is possible to obtain its implementation. The need of a synchronous channel in order to have timeliness guarantees limits its utilization. However, as we have already said, we only need a small bandwidth channel for control information, not generic data.

In this paper, we proposed access control

mechanisms to handle that problem. Depending on the required application timeliness granularity, we can use control mechanisms on top of the existing MAC (software approach), or we can build a simple interceptor (hardware approach) to enforce a deterministic collision resolution at low level.

Although the work presented here is more communications oriented, we are also pursuing the addressing of aspects related to processing, which are needed in order to build a full system.

References

[1] Carlos Almeida and Paulo Veríssimo. An adaptive real-time group communication protocol. In *Proceedings of the First IEEE Workshop on Factory Communication Systems*, Leysin, Switzerland, October 1995.

[2] Carlos Almeida and Paulo Veríssimo. Timing failure detection and real-time group communication in *quasi-synchronous* systems. In *Proceedings of the 8th Euromicro Workshop on Real-Time Systems*, L' Aquila, Italy, June 1996. Also available as INESC technical report RT/20-95.

[3] Carlos Almeida and Paulo Veríssimo. Using lightweight groups to handle timing failures in *quasi-synchronous* systems. In *Proceedings of the 19th IEEE Real-Time Systems Symposium*, Madrid, Spain, December 1998.

[4] Carlos Almeida, Paulo Veríssimo, and António Casimiro. The quasi-synchronous approach to fault-tolerant and real-time communication and processing. Technical Report CSTC RT-98-04, Instituto Superior Técnico, Lisboa, Portugal, July 1998.

[5] Martin de Prycker. *Asynchronous Transfer Mode: Solution For Broadband ISDN (Third Edition)*. Number ISBN 0-13-342171-6. Prentice Hall, 1995.

[6] ISO. *ISO DIS 8802/3-85, Carrier Sense Multiple Access with Collision Detection*, 1985.

[7] G. LeLann. The 802.3d protocol: A variation of the ieee802.3 standard for real-time lans. Technical report, INRIA, France, 1987.

[8] G. LeLann and N. Rivierre. Real-time communications over broadcast networks: the csma-dcr and the dod-csma-cd protocols. Technical Report Research Report 1863, INRIA, France, June 1993.

[9] J. Rufino and P. Veríssimo. A study on the inaccessibility characteristics of ISO 8802/4 Token-Bus LANs. In *Proceedings of the IEEE INFOCOM'92 Conference on Computer Communications*, Florence, Italy, May 1992. IEEE. also INESC AR 16-92.

[10] J. Rufino and P. Veríssimo. Minimizing token-bus inaccessibility through network planning and parameterizing. In *Proceedings of the EFOC/LAN92 Conference*, Paris, France, June 1992. IGI. also INESC AR 17-92.

[11] José Rufino and Paulo Veríssimo. MATRIOT: a MAC protocol interceptor optimizes Token-Bus accessibility. Technical Report RT/xx-93, INESC, Lisboa, Portugal, 1993.

[12] C. Venkatramani and T. Chiueh. Design, implementation, and evaluation of a software-driven real-time ethernet protocol. In *Proceedings of the ACM SIGCOMM 95*, 1995.

[13] P. Veríssimo, J. Rufino, and L. Rodrigues. Enforcing real-time behaviour of LAN-based protocols. In *Proceedings of the 10th IFAC Workshop on Distributed Computer Control Systems*, Semmering, Austria, September 1991. IFAC.

[14] Paulo Veríssimo and Carlos Almeida. Quasi-synchronism: a step away from the traditional fault-tolerant real-time system models. *Bulletin of the Technical Committee on Operating Systems and Application Environments (TCOS), IEEE Computer Society*, 7(4):35–39, Winter 1995.