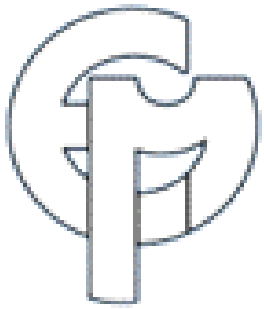
RTLIA 2002

1st Intl. Workshop on Real-Time LANs in the Internet Age

(Satellite Event to ECRTS'02)

Pre-prints

Technical University of Vienna, Austria
June 18th, 2002



Message from the Workshop Chair

Welcome to the 1st international workshop on Real-Time LANs in the Internet Age (RTLIA), a satellite event of the Euromicro Conference on Real-Time Systems, held in Vienna, Austria, 18th June 2002. We sincerely hope that this is a starting point to a successful series that will have its second edition, also as a satellite event of the 2003 edition of the Euromicro Conference on Real-Time Systems, to be held in Porto, Portugal, next year.

In the recent past, works addressing LANs for RT distributed computer-controlled applications could more or less start like this: "... despite its popularity, low-price, maturity and stability, Ethernet has a serious drawback concerning the support of real-time distributed applications, since it is difficult to guarantee predictable delays in delivering messages to network nodes ...". It is not like this anymore. Advances in switched-Ethernet are just one example of technology that is re-shaping the way RT distributed systems are engineered.

In fact, advances in networking and information technologies are transforming real-time concerns into a mainstream activity spanning over almost every computer system. It is now recognised that future computer systems will be intimately tied to real-time computing and to communication technologies.

For this vision to succeed, complex heterogeneous networks (including mobile/fixed and wireless/wired) need to function in a predictable, flawless and interoperable way. If we add on top of these arguments the thought that in a near future, all networks may be operating with some equivalent of IP technology, we may come to a point in the time where, theoretically, LANs, PANs (Personal Area Networks) and WANs may likely to converge...

The goal of this workshop is to bring together people from industry and academia that are interested in all aspects of using commodity LAN technologies to support real-time and dependable applications in the Internet Era. The workshop will provide a relaxed forum to present and discuss new ideas, new research directions and to review current trends in this area.

We tried to set up the most attractive program the possible, grouping contributors into a mix of short presentations, panel discussions and positions statements. The goal was to try to foster the participants' interaction, rather than focusing on formal presentations followed by the usual couple of minutes for questions.

I would like to acknowledge all participants, authors, reviewers, and the organisation committees (both RTLIA and ECRTS), that have made this event a successful one.

Eduardo Tovar,
Polytechnic Institute of Porto, Portugal

RTLIA2002 Workshop Chairman:

Eduardo Tovar
Polytechnic Institute of Porto, Portugal
email: emt@dei.isep.ipp.pt

RTLIA2002 Organising committee:

Eduardo Tovar
Polytechnic Institute of Porto, Portugal
emt@dei.isep.ipp.pt

Luis Miguel Pinho
Polytechnic Institute of Porto, Portugal
lpinho@dei.isep.ipp.pt

Thilo Sauter
Technical University of Vienna, Austria
sauter@ict.tuwien.ac.at

Gerhard Pratl
Technical University of Vienna, Austria
pratl@ict.tuwien.ac.at

ECRTS2002 General Chairman:

Peter Puschner
Technical University of Vienna, Austria
peter@vmars.tuwien.ac.at

ECRTS2002 Program Chairman:

Gerhard Fohler
Maelardalen University, Sweden
gerhard.fohler@mdh.se

RTLIA2002: http://www.hurray.isep.ipp.pt/rtlia2002/

ECRTS2002: http://www.idt.mdh.se/ecrts02/

Technical Program

9:00 - 9:15 **Welcome Address**

9:15 - 10:15 **Session 1: Paradigms for Control Networks**

Session Format:

4 presentations of 15 minutes each.
Author of paper 1 starts by introducing a vision. Then, authors of papers 2 and 3 will have 15 minutes each to give a more traditional view of the traditional networks. Finally, there will be a talk of 15 minutes, based on papers 4 and 5, presenting an approach to bridge the two worlds.

Session Position Papers:

1. *A Virtual Global Bus Active Messaging Protocol for Sensor Webs*
   D. Andrews, J. Evans
   University of Kansas, USA
2. *Timing-Independent Safety on Top of CAN*
   G. Lima, A. Burns
   University of York, United Kingdom
3. *Byzantine Fault Containment in TTP/C*
   G. Bauer, H. Kopetz, W. Steiner
   Vienna University of Technology, Austria
4. *Internet - Technologies that are Missing*
   P. Cach, P. Fiedler
   Brno University of Technology, Czech Republic
5. *Ethernet Interface in Application - a case study*
   P. Cach, P. Fiedler
   Brno University of Technology, Czech Republic

10:15 - 10:45 **Session 2: Which future for traditional control networks?**

Session Format:

Panel discussion of 30 minutes, involving all the audience and particularly authors of papers of session 1.
Authors of papers 2 and 3 to start the "hostilities".

10:45 - 11:15 **Coffee Break**

11:15 - 11:45 **Session 3: An Emerging Control Network: Switched-Ethernet**

Session Format:

3 presentations of 10 minutes each.
Author of paper 6 gives an overview. Then, author of paper 7 addresses specifically Ethernet/IP while author of paper 8 gives a talk focusing on IDA.

Session Position Papers:

6. *Real-Time with Ethernet*
   R. Messerschimdt
   Otto-v.-Guericke-Universität, Magdeburg, Germany
7. *Utilization of Modern Switching Technology in EtherNet/IP™ Networks*
   A. Moldovansky
   Rockwell Automation, USA
8. *Ethernet based Realtime LAN for Automation Applications*
   M. Buchwitz
   Jetter AG, Germany

11:45 - 12:15 **Session 4: An Emerging Control Network: Shared-Ethernet?**

Session Format:

3 presentations of 10 minutes each.
Author of paper 9 addresses a probabilistic approach for providing RT behaviour. Then, author of paper 10 suggest the use of a token-passing procedure on top of shared-Ethernet, while author of paper 11 will talk of providing time triggered mechanisms to Ethernet-based networks.

Session Position Papers:

9 *Fuzzy Traffic Smoothing: another step towards Statistical Real-Time Communication over Ethernet Networks*
R. Caponetto, L. Lo Bello, O. Mirabella
Università di Catania, Italy

10 *A Multipoint Communication Protocol based on Ethernet for Analyzable Distributed Real-Time Applications*
J. Martinez, M. Harbour, J. Gutierrez
University of Cantabria, Spain

11 *Flexibility, Timeliness and Efficiency in Ethernet*
P. Pedreiras, L. Almeida
University of Aveiro, Portugal

12:15 - 12:45 **Session 5: Why Moving to Switched-Ethernet?**

Session Format:

Panel discussion of 30 minutes, involving all the audience and particularly authors of papers of session 3 and session 4.
Authors of papers in session 4 to start the "hostilities".

12:45 - 14:30 **Lunch**
We have booked in the Restaurant Wieden Bräu. Participants are free to choose alternative place.

14:30 - 15:30 **Session 6: Supporting Real-time Applications with Ethernet**

Session Format:

4 presentations of 10 minutes, each one followed by 5 minutes for discussion.

Session Position Papers:

12 *SBM protocol for providing real-time QoS in Ethernet LANs*
A. Koubaa, A. Jarraya, Y.-Q. Song
LORIA, France

13 *Deadline First Scheduling in Switched Real-Time Ethernet - Deadline Partitioning Issues and Software Implementation Experiments*
H. Hoang, M. Jonsson, A. Larsson, R. Olsson, C. Bergenhem
Halmstad University, Sweden

14 *Designing, Modelling and Evaluating of Switched Ethernet Networks in Factory Communication Systems*
N. Krommenacker, J.P. Georges, E. Rondeau, T. Divoux
CRAN-CNRS, France

15 *Real Time on Ethernet using off-the-shelf Hardware*
J. Loeser, H. Hartig
TU Dresden, Germany

15:30 - 15:45 **Break**

15:45 - 16:15 **Session 7: Wireless Networks**

Session Format:

2 presentations of 10 minutes, each one followed by 5 minutes for discussion.

Session Position Papers:

16 *Bluetooth - One of the best WPAN solutions for bridging PAN and wider networks?*
T. Dulai, A. H. Medve
University of Veszprém, Hungary

17 *An Experimental Testbed for Using WLANs in Real-Time Applications*
T. Lunheim, A. Skavhaug
NTNU, Trondheim, Norway

16:15 - 16:45 **Coffee Break**

16:45 - 17:25 **Session 8: Networking Support of Streamed-data Applications**

Session Format:

4 presentations of 10 minutes each. Firstly a talk covering real-time issues in distributed multimedia applications based on papers 18 and 19. Then a presentation of peer-to-peer networking aspects (paper 20). Then a talk on ATM-based infrastructures. Finally a talk about novel approach for routing IP packets.

Session Position Papers:

18 *Workload Balancing in Distributed Virtual Reality Environments*
M. Ditze, F. Pacheco, B. Batista, E. Tovar, P. Altenbernd
C-Lab, Germany; Polytechnic Institute of Porto, Portugal

19 *Common Issues in Real-Time and Media Processing*
P. Altenbernd, M. Ditze
C-Lab, Germany

20 *Distributed Video-on-demand Services on Peer-to-Peer Basis*
C. Loeser, P. Altenbernd, M. Ditze, W. Mueller
C-Lab, Germany

21 *Dynamic Real-Time Bandwidth Sharing Algorithm for Broadband Multimedia Communication Systems*
Y. Atif
United Arab Emirates University, UAE

22 *Synchronous Time Division Internet for Time-Critical Communication Services*
T. Yakoh
Keio University, Japan

17:25 - 17:45 **Session 9: Communication Infrastructure for Distributed Streamed-Data**

Session Format:

Panel discussion of 20 minutes, involving all the audience and particularly authors of papers of session 8.
Author of paper 22 to start the "hostilities": do we need to modify the existent routing technology?

17:45 - 18:30 **Session 10: Converging Infrastructure Combining PANs/LANs/MANs/WANs**

Session Format:

This session will consist of presentation of paper 23 (15 minutes) followed by a panel discussion involving all the audience on the topic: will it be possible to have a converging global infrastructure combining PANs/LANs/MANs/WANs?

Session Position Papers:

23 *Convergence*
O. B. Madsen, J. Dalsgaard, H. Schiøler
Aalborg University, Denmark

20:00 **Dinner**
Arrangements for a dinner will be provided. However participants are free to choose alternative place.

Table of Contents

A Virtual Global Bus Active Messaging Protocol for Sensor Webs

David Andrews, Joe Evans
University of Kansas

dandrews@ittc.ukans.edu, evans@ittc.ukans.edu

Mixed signal, micro-electro-mechanical, and wireless communication technologies have accelerated our ability to define and build existing and newly emerging hardware platforms into globally and geographically distributed virtual computing systems. The application domain of these new sensor webs is broad, ranging from bio-medical applications, through remote environmental analysis and sensing, to bio-terrorism. Operationally, these future sensor web systems will be comprised of hundreds of thousands of small, autonomous devices that dynamically engage or disengage in the generation of data, and processing of the data into knowledge. Nodes will be deployed in an ad-hoc fashion, with no a-priori knowledge of network and sensor/actuator connectivity. The physical structure of each node will be designed to minimize power, prolong useful deployment time, increase reliability through large numbers, and efficiently process the low level operational mode dictated by the sensors. Behaviorally, node selection, data generation, and signal pre-processing will occur dynamically in order to allow unknown numbers and placement of sensors to be controlled in a system centric coordinated fashion, and increased system reliability.

A challenge exists in developing appropriate network centric models that tightly integrate the computing and communications requirements to "enable (a) accurate distributed sensing, (b) multimode data fusion, (c) transformation from one domain to another, (d) extraction of key information, and (e) detection and circumvention of faulty sensors in ultra large arrays"[1]. To support this challenge we have been exploring new models that

1) Define a new integrated communications/computation machine model for sensor web processors in terms of a Meta instruction set architecture
2) Define a new link layer protocol as a part of the computation/communication fabric that forms a virtual global bus independent of exact sensor locations, types, or numbers. We are investigating this approach as it takes advantage of the law of large numbers in increasing system connectivity, while eliminating power intensive multi-hop route explorations and maintenance of routing tables in sensor node memories.

First, we present background on sensor webs in order to understand their operational domain. After presenting this background, we survey the current state of the art in developing wireless sensor web networks and propose a radically different computation/communication model along with a new supporting link layer protocol.

Requirements for Sensor Web Systems

Conceptually, evolving sensor web systems can be viewed as functional info-spheres of computation, where hundreds of thousands of sensors are data producers, distributed autonomous agent processing algorithms are knowledge generators, timeliness responses are specified on dynamic, non-stochastic knowledge states, and distributed actuators co-ordinate to achieve an enhanced system response. Culler [2] characterizes this new genre of embedded software as being agile, self-organizing, critically resource constrained, and communication-centric on numerous small devices operating as a collective. The operational mode is intensely concurrent, with environmentally bursty activity. The burst rate is projected to be very low, in the 1-10 hertz range. The application space is ubiquitous, spanning numerous devices that interact in a context aware manner. Additionally, power considerations will be a driving factor in the realization of these systems. Current goals include 5 mW/MIP power factors, orders of magnitude

lower than current techniques and technology can provide.

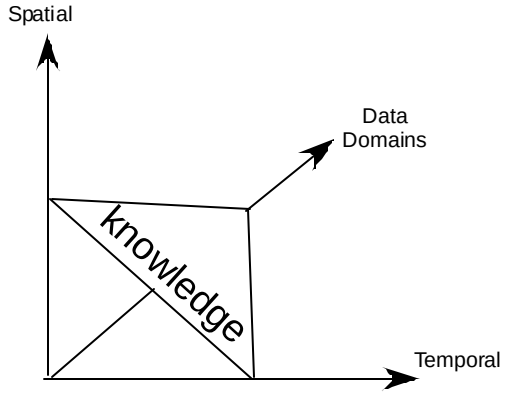


Figure 1. Course Decomposition of Embedded Systems Functional Domains

The quality of a response is both a function of the timeliness, and the quality of knowledge at an instant in time. We can evaluate this model by a course decomposition of the abstract domains that comprise this model as shown in Figure 1.

Computational nodes perform demand driven pass through conditioning of sensor data, and engage in dynamic knowledge formation via data sharing and processing across distributed platforms. This represents a data centric model, with focus on applying operations to data fields and sets. The formation of knowledge is based on retrieval and combination of data across the network in a spatial domain, and timeliness of data updates. Primitive operations, such as domain data selection, reduction and combination are required to support processing within this new paradigm. This is fundamentally opposite to non-embedded models that have "flat" spatial locality and no temporal or domain specific planes. Temporal requirements result from sensor input rates as well as timeliness constraints on the knowledge production and response. The computational model should elevate timeliness information up into the application domain where the relationship between time, precision, and quality of service are best understood.

Sensor Web Architectures

Researchers are investigating clustering and domain selection approaches that allow individual sensors to engage and disengage from the network in a power aware fashion. The sensor information networking architecture (SINA) [3] forms hierarchical clusters of autonomous groups. This clustering process is applied recursively to form the hierarchy. Information is accessed via attribute-based naming instead of explicit addressing. Nodes are queried for information based on attributes, where complex queries can be formed with little overhead within the network. In contrast to SINA, a self-organizing medium access control for sensor networks (SMACS) [8] has been proposed that enables nodes to discover neighbors without the need for master nodes. The SMACS architecture builds a flat topology with no clusters or cluster heads. The eavesdrop and register (EAR) [8] architecture has been developed for communication between mobile nodes and stationary nodes on the ground. To conserve energy, mobile nodes keep a registry of all sensors within a neighborhood and make handoff decisions whenever the SNR drops below a pre-determined threshold value. During a bootup period, invitation messages are broadcast as a trigger. Each mobile node eavesdrops and forms a registry of all stationary nodes within hearing range.

Routing Protocols

Most proposed protocols fall into two main categories: flat routing protocols, and hierarchical protocols. The objective of all routing protocols is to limit node to node communications between pairs of near nodes to reduce power.

Flat Routing Protocols

The first flat routing protocol, sequential assignment routing (SAR) [3], builds multiple routes between a sink and source. This is to minimize the time and cost of computing new routes during failures. A routing tree is built outwards from the sink nodes that attempt to minimize the use of low QoS and energy reserves. Each node belongs to multiple paths and each sensor can control which one-hop neighbor of the sink to use for messaging. The SAR algorithm uses an adaptive QoS metric and a measure of energy resources to arrive at an additive QoS metric and a weight coefficient associated with a packet priority level. During system operation, the SAR algorithm attempts to minimize the average weighted QoS metric. This algorithm requires re-computing paths to account for changes in network topology.

Directed diffusion is a flat routing protocol proposed by Estrin et. al. [4] that allows sensor data to be named. Sink nodes query the sensor web based on data names, and sensor nodes may

selectively respond. Intermediate nodes may route data from sensors towards the sink node. Ye et. al. [9] proposed a minimum cost forwarding algorithm for large sensor networks. In this approach, each node contains a least cost estimate between itself and the sink node. Each message in the system is broadcast, and intermediate nodes check to see if they are on the least cost path. If so, the node re-broadcasts the message. Kaulik et. al. [3] proposed the sensor protocols for information via negotiation (SPIN). These protocols disseminate individual sensor information to all sensor nodes under the assumption that all are potential sinks. The solutions use negotiation and information descriptors to overcome the potential information implosion that can be caused by flooding the network with messages sent to all nodes.

Hierarchical Routing Protocols

Chandrakasan et. al. [6] proposed a low energy adaptive clustering hierarchy (LEACH) routing protocol as an energy efficient communication protocol for wireless sensor networks. In LEACH, self-elected cluster heads collect data from all sensor nodes in a cluster, and use data fusion methods to aggregate and transmit the data directly to the base station. The appointment of a cluster head is made periodically with the self appointed cluster head announcing it's role to it's neighbors.

Proposed Link Layer Protocol

Our system approach shares some behavioral commonality with flat routing protocols such as those used in directed diffusion and SPIN. However, we choose to investigate a slightly different approach that seeks higher reliability and greater coverage at what will hopefully result in modest power requirements. Our approach extends the basic principle of Active Messages in TinyOS into a network centric perspective by integrating handler identifiers into the routing of messages at the link layer level. We form communications across the network by utilizing coherent fusion of transmission energies of the sensor nodes to form a virtual global bus. We chose this approach in order to address two unsolved issues. First, we believe that exploring multi hop routes and storing routing information is inherently time and power consuming, just the opposite of what is desired. With the current approaches, receivers are allocated time slots and are allowed to sleep during the intra-periods between allocated slots. This can result in missed messages and coverage. Second, multi-hop routes introduce single point of failure potentials. A multi-hop organization requires the system to constantly perform self-diagnosis and exploration of healthy nodes for new routes. Figure 6 shows the approach we chose to investigate. In this approach, a sender node arbitrates for the bus (as discussed below) and once obtained, continues with transmission of data. All other nodes that are monitoring then assume relay status, and repeat the transmission sequence from the sender. This approach will impose longer data width sizes on the bus. However, prior research indicates that the amount of data being transmitted is relatively modest, allowing longer data cell times. We plan on investigating this tradeoff during the proposal. Finally, we believe our approach is a continued step in the direction of forming a tight computation/communication model. This model allows attributed based access of unknown numbers of sensor nodes, providing flexibility in ad-hoc deployment and operation.

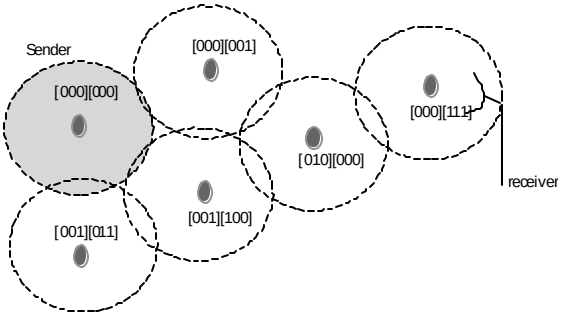


Figure 6. Coherent Domain Coverage

Arbitration/Protocol

Our protocol is similar to the CAN [7] bus in arbitration and protocol but with identifiers serving as handler identifiers ala Active Messaging. This allows fully distributed arbitration between nodes attempting to become bus master. In the CAN protocol, an approach similar to open collector circuits is used to control the logic level on the bus. Multiple masters attempt to drive their identifier onto the bus while also monitoring the success of each bit from the identifier. A logic "0" is dominant, and all nodes attempting to drive a 1 when a logic 0 is present immediately transition to slaves. Within our approach, when a device determines that it is not the bus master, it transitions to a "repeater" device that coherently transmits the data bits that follow

on the bus. Communications across the bus result from explicit remote transfer requests on a particular identifier, or may also be initiated by a node when desiring to broadcast data to other listening devices.

Assignment of Identifiers

Identifiers are associated with knowledge domain handlers. Meta (ISA) instructions are defined within a specific domain. The protocol controller matches handler identifiers streaming on the bus to a domain handler match register. The match register determines if a handler is resident on the node. The advantage of this protocol is that it allows instructions, queries, and data to be broadcast within a knowledge domain. A query can easily be issued on an identifier soliciting sensor nodes to transmit data within the knowledge domain.

We have defined Meta Instructions within our ISA that are network operations, such as gather, broadcast and reduction. The gather operation will perform bundling to reduce the overall bandwidth requirements, and the reduction operator will specify an arithmetic or Boolean operation to be applied to the data within the network. We are currently investigating implementation techniques for supporting these instructions across the network. Included in our investigation will be adopting current techniques such as frequency and time multiplexing in order to increase the available bandwidth. On advantage of our approach is that logically it can be implemented across a wide range of physical networks and protocols. Additionally, we will investigate the power tradeoffs between forming coherent bursts, and updating and maintaining routing tables.

Conclusion

In this paper, we have presented the operational mode of emerging sensor web systems. Sensor web nodes will be required to operate autonomously, adapting to ad hoc deployment of thousands of nodes in unpredictable deployment patterns. Of concern within these systems are the network organization and link layer protocol. The link layer protocol must support power conserving operation while still providing reliable communications. Current approaches adopt TDMA based techniques that minimize power draw by shutting down nodes for intra-transmission periods. While this approach minimizes power, it does not encourage higher reliability of connections between multi-hop nodes, and requires power dissipation in path exploration and forming of clusters. We propose a new approach based on the CAN protocol, which offers the promise of higher reliability and greater connectivity. Our link layer protocol is a portion of a new computation/communication integrated machine model for sensor web systems.

Bibliography

[1] http://www.nsf.gov/pubs/2002/nsf02039/nsf02039.html

[2] David Culler et. al, A Network-Centric Approach to Embedded Software for Tiny Devices, Proceedings of the First International Workshop, EMSOFT 2001, Tahoe City, Ca, Oct. 2001

[3] Rentala, Praveen, Musunuri, Ravi, Gandham, Shashidhar, Saxena, Udit, "Survey on Sensor Networks"

[4] Chalermek Intanagonwiwat, Ramesh Govinda, and Deborah Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks" Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom'2000), Boston, Massachusetts, August 2000.

[5] Ye F. Chen, A., Liu, S. Zhang L. "A scalable solution to minimum cost forwarding in large sensor networks" Proceedings of the Tenth International Conference on Computer Communications and Networks, pp. 304-309, 2001

[6] Wang, A. Chandrakasan, A., "Energy efficient system partitioning for distributed wireless sensor networks", IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 2. pp. 905-908, 2001

[7] http://www.can.bosch.com/index.html

[8] Sohrabi, K., Gao, J. Ailawadhi, V. Pottie, G.J."Protocols for self-organization of a wireless sensor network" IEEE Personal Communications, Vol. 7., Issue 5, Pages 16-27, 2000

[9] Ye F. Chen, A., Liu, S. Zhang L. "A scalable solution to minimum cost forwarding in large sensor networks" Proceedings of the Tenth International Conference on Computer Communications and Networks, pp. 304-309, 2001

Timing-Independent Safety on Top of CAN

George Lima* Alan Burns

Real-Time Systems Research Group
Department of Computer Science, University of York,
Heslington, York, England, YO1 5DD
{gmlima,burns}@cs.york.ac.uk

Abstract

We describe an approach to designing CAN-based distributed real-time systems so that safety is preserved regardless of timeliness. Our approach offers gains with respect to both fault tolerance and flexibility aspects and so it is attractive to support those systems that have critical tasks (e.g. control systems) and at the same time are connected to non-predictable networks (e.g. the Internet).

1. Introduction

The correctness of real-time systems is specified in terms of both safety and timeliness. The safety requirement has led to the use of distributed platforms to implement fault tolerance mechanisms. Making a system distributed consists of spreading the processes that carry out its computation across different machines linked to each other by means of a communication network. In turn, the timeliness requirement has made the *synchronous* model of computation a natural choice for implementing distributed real-time systems. According to this model *all* sent messages arrive within a known interval of time (communication is synchronous) and *all* computation is finished within a bounded time (processing is synchronous).

Assuming synchronous processing in real-time systems is reasonable since bounds on processing times can be derived by carrying out appropriate schedulability analysis. However, assuming synchronous communication may be a restriction. Indeed, the communication network is a critical component of distributed systems since it is a shared resource and is most subject to transient faults and overload conditions. As systems that are based on the synchronous model use the knowledge about the assumed bounds to guarantee safety, we say that they are *timing-dependent safe*. For example, if a message that is supposed to be received within a given interval of time does not arrive, the receiver process may conclude that the sender is faulty. However, if the message is just late, the result of the computation by the receiver may be inconsistent (i.e. unsafe).

In this work we demonstrate that it is possible to build timing-independent safe hard real-time systems on top of the Controller Area Network (CAN) [2]. CAN is a broadcast network that is widely used in the implementation of distributed hard real-time systems. The idea is to make co-operating processes *agree* on their computations by exchanging messages. As we will see, due to the message scheduling and error-recovery mechanisms of CAN this can be done straightforwardly. The benefits of our approach can be verified by considering a *semi-synchronous* model of computation based on CAN properties. This model relaxes the communication synchronism until a point beyond which the system's timeliness would be compromised. Moreover, the described approach is particularly interesting due to its flexibility since by using it systems may tolerate unpredictable behaviour caused by overload or faulty scenarios in some nodes of the system. These characteristics make the approach attractive, mainly for supporting those systems that have critical tasks (e.g. control systems) and at the same time are connected to non-predictable networks (e.g. the Internet).

2. Model of Computation

In this section we define a semi-synchronous model of computation having CAN as the communication network. As our main goal is to show that one can design timing-independent safe protocols using CAN, this

*Supported by CAPES/Brazil under grant: BEX1438/98-0.

model allows message timing/omission faults to take place.

2.1. Processing Model

We consider systems made of geographically distributed nodes, which are fully connected to each other by means of a CAN-based communication network. Each process is allocated to a node. Processes communicate to each other only by exchanging messages across the network. Processes may only fail by crashing. Correct processes are those that never crash. If a process crashes at time t, it stops both sending and receiving messages indefinitely from time t (i.e. crashed processes do not recover).[1]

Processes may perform local and non-local tasks. Local tasks are those that do not depend on the communication network (i.e. message-send or message-receive events). We assume synchronous processing, by which we mean that the worst-case response times of local tasks are known. This can be guaranteed in practice by applying real-time scheduling techniques [1].

2.2. Communication Model

The assumed communication network is typified by the Controller Area Network (CAN) [2]. Due to its deterministic collision resolution based on priorities and the built-in error-recovery schemes, CAN is widely used for supporting hard real-time systems. Indeed, CAN provides a very resilient error-detection and recovery mechanism that can handle most failures consistently. Hence, we assume that messages cannot be either arbitrarily created or corrupted by the network.

Errors on CAN are detected by the transmitter or receiver nodes while monitoring the transmission of messages on a bit-by-bit base. If a message is detected corrupted, it is scheduled for re-transmission according to its priority. Although this error recovery and the message scheduling schemes used in CAN provide a high degree of reliability and predictability, they may lead to some inconsistency is some specific cases. In fact, it has been shown that in some scenarios (involving the last but one bit of the transmitted message) a set of receivers can accept a given transmitted message while others reject it [4, 3]. In this situation three inconsistent scenarios may take place: (a) if the transmitter crashes after the detection of the error and before the re-transmission, its transmitted message will be inconsistently omitted at some nodes; (b) if the transmitter does not crash, it re-transmits the message and so some receivers will receive the message more than once; and (c) this scenario has the same effect as (a) and happens if the transmitter does not crash but it does not detect the faulty transmission [3]. Notice that scenario (a) is associated to process crashes while (b) and (c) are due to the way error-recovery is carried out in CAN. According to some simulations [4, 3], the probability of occurrence for these inconsistent scenarios varies between 8.80×10^{-3} and 3.96×10^{-8} per hour. Although these scenarios are unlikely, they have to be considered when dealing with critical applications.

Based on the characteristics of CAN described above, we assume that messages may be dropped by the network (due to the inconsistent scenarios) or arbitrarily delayed by the network (due to CAN message scheduling mechanism). However, in the absence of the inconsistent scenarios CAN provides what is called *atomic broadcast* [4, 3]: transmitted messages are totally ordered and received either by all correct processes or by none. As we will see in the next section, this property can be used to design timing-independent safe systems.

3. Timing-Independent Safety

Ideally, systems must be safe regardless of the present level of synchronism. This means that processes must only take decisions during their computation based on their view about the whole system, instead of on the time. It is clear that such an approach does not work in general. The following example illustrates this.

Example 3.1. *Two processes, p and q say, are cooperating throughout their execution. Suppose a moment during the execution of p when it is waiting for a message from q in order to take a decision in accordance with q's computation. As process p eventually has to make progress (i.e. it has to meet deadlines), it cannot wait forever (neither can q). Hence, there may be a moment at which p has to make progress regardless of q's message. If some fault prevents q's message from being delivered at p, p may violate safety. If q is crashed, though, p is free to take its own decision.*

The example above illustrates a dilemma between safety and timeliness: favouring one may compromise the other. Notice that the reason behind this dilemma is that it is impossible for processes to have reliable information about failures of remote processes if the synchronous model is not assumed. A tradeoff between safety and timeliness, though, can be achieved by considering other kinds of synchronism. For instance, if

[1] A protocol for re-introducing recovered processes could be added but this is beyond the scope of this work.

the system provides atomic broadcast, it is possible to implement the system so that no inconsistent decision can be taken. Making use of this atomic broadcast primitive, our illustrative example can have the following solution. After waiting for the message from q, p atomically broadcasts a message to pass on the decision on its computation. After receiving its own message, p knows that if q is not faulty, it will also receive the same message and in the same order so that q will also make progress according to p's message. Therefore, both processes will be safe regardless of the time messages take to be delivered.

As we have seen, however, CAN does not provide perfect atomic broadcast due to some inconsistent scenarios. Hence some extra effort has to be made. Indeed, our approach to building timing-independent safe systems on top of CAN requires that co-operating processes execute an agreement phase during their computation to ensure safety. During this phase processes exchange messages in order to reach the same view about the system despite scenarios (a), (b) and (c). Notice that by assumption (section 2.2) if a transmitted message is received by a process and scenarios (a), (b) and (c) do not take place, then this message is received by all correct processes. In order to take these scenarios into account we assume that no more than f inconsistent scenarios may take place during the agreement phase. As we have seen, the probability of these scenarios taking place, although not negligible, is not significantly high. Thus, one can choose a value for f that is suitable for the targeted system. In the next sections we discuss the safety, timeliness and flexibility aspects.

3.1. Ensuring Safety

Assume that there is up to f inconsistent scenarios. If any message is received by some process and scenarios (a), (b) and (c) do not take place, the message is also received by all correct processes (by the CAN properties). As a process that receives a message does not know whether or not other processes also received this message, it has to re-transmit the message f times. After the reception of the last re-transmission the process knows that all correct processes also received the message (at least once). Hence, up to $f+1$ transmissions by each process are necessary to guarantee the reception of the message by all correct processes. This is the basic idea of the agreement phase and is described in the algorithm of figure 1. In other words, lines 1-7 of the algorithm can be inserted into the normal code of any critical task of processes that co-operate.

The agreement phase consists of up to $f+1$ rounds of

```
    /* ... normal computation ... */
    /* m contains the result of the computation */
(1) m.k ← 0
(2) while m.k < f + 1 do
(3)     broadcast(m)
(4)     wait for [ receive m' such that m'.k ≥ m.k ]
(5)     get the first received m' such that m'.k ≥ m.k
(6)     m ← m'; m.k ← m'.k + 1
(7) endwhile /* ... processes agree on m ... */
```

Figure 1. The agreement phase algorithm

message exchanges. Any exchanged message is tagged with an integer counter that is used to keep track of the number of rounds seen by the processes. At the end of this phase, all correct processes agree on the same message so that they can make progress based on the same view of the system. Notice that there is no reference to time in the agreement phase. Safety is ensured just by message exchanging. For the sake of illustration, consider example 3.1 and suppose that $f = 1$. Two possible executions of p and q are shown in figure 2. The numbers along the time line represent the values of m.k at each process. In A, process p does not receive any message from q. This may be due to a crash fault or asynchrony between the executions of p and q, say. Then, p sends its message twice during its agreement phase. If q is not crashed, it receives at least one transmission consistently. During its execution q eventually picks up the message sent by p and takes the decision on its computation accordingly in order not to violate safety. In execution B, an inconsistent scenario takes place. The message from q is not received by p but is received by q. After the second transmission, though, both p and q choose the same message (q's message).

It is important to emphasise that this simple agreement protocol does not guarantee atomic broadcast. Messages are still being delivered out of order. This agreement phase is enough, however, to ensure safety regardless of time. Indeed, in the example B p gives up its own message to accept q's.

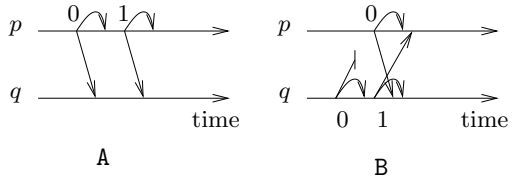


Figure 2. Illustration of the agreement phase

3.2. Ensuring Timeliness

A real-time system is made of several services, which may have different priorities. These priorities are usually assigned according to the urgency of execution. Thus, the higher the priority of the service, the higher the priority of the messages sent by its processes. A system characterised in such a way makes the analysis of its feasibility straightforward. For example, one can carry out well known schedulability analysis for this purpose (e.g. [1, 5]). Indeed, the approach discussed in this work does not require any special mechanism to check the system timeliness.

For example, consider a distributed system made of three services, `H` (highest priority), `M` (medium priority) and `L` (lowest priority). The worst-case scenario in each node is determined (as usual) when all tasks of the node are released at the same time. In this situation, messages sent by service `M` and `L` may be transmitted only after messages sent by `H`. Now, in order to illustrate the strength of our approach assume that in the absence of the inconsistent scenarios just the highest priority message arrives at all its destinations within a known bounded delay. This assumption may represent a given worst-case scenario due to possible transient faults in the network, say. Even with this low level of synchronism in the communication network one is assured that safety is not violated. Yet, to derive timeliness, the message scheduling provided by CAN guarantees that after `H` finishes sending messages, `M` can make progress and so on. If all three services meet their deadlines, we say that timeliness is ensured.

Clearly, some considerations regarding the application has to be taken into account when analysing the system timeliness. For example, if it is known that processes p and q, say, of a given service start executing their tasks approximately at the same time in different nodes, some tightness guarantee between their computation can be derived. However, it is important to emphasise that the guarantee of safety does not need any reference to time whatsoever. In other words, the dynamics of the system dictate the time spent by its computation and can be derived by analysing the system after knowing that its safety is not violated.

3.3 The Flexibility Aspect

It is important to note that considering safety and timeliness independently brings flexibility for systems with respect to both the communication synchronism and the processing synchronism. Consider a typical application which has hard real-time tasks distributed across a set of nodes, say. It would be useful if information about the system could be remotely monitored. Doing this using fieldbus networks (such as CAN) may not be viable due to their low bandwidth. Hence, the monitoring tasks (well modelled as soft tasks) might use non-predictable communication networks (such as the Internet), which in turn might overload the nodes in which such tasks run (since TCP connections may introduce unpredictable delays). If the system is designed in line with the timing-independent safety approach, one can avoid the unpredictable behaviour of the monitoring system (soft tasks) interfering in the critical tasks. For instance, even if one of the processes in figure 2 is subject to these overload conditions (since it may be running in the same node as the monitoring system), the monitoring system will never present inconsistent information.

4. Conclusion

The problem of designing timing-independent safe real-time systems has been addressed. As we have seen, CAN offers powerful properties that can be used to achieve such an objective. In general, the approach discussed in this work is very attractive due to its simplicity and can be used to enhance both the fault tolerance and the flexibility of real-time systems.

References

[1] A. Burns and A. Wellings. *Real-Time Systems and Programming Languages.* Addison-Wesley, 3nd edition, 2001.

[2] Int'l Standards Organisation. *ISO 11898. Road Vehicles – Interchange of digital information – Controller area network (CAN) for high speed communication*, 1993.

[3] J. Proenza and J. Miro-Julia. MajorCAN: A Modification to the Controller Area Network Protocol to Achieve Atomic Broadcast. In *IEEE Int'l Workshop on Group Communication and Computations (IWGCC 2000). Taipei, Taiwan*, Apr. 2000.

[4] J. Rufino, P. Veríssimo, G. Arroz, C. Almeida, and L. Rodrigues. Fault-tolerant broadcasts in CAN. In *Symposium on Fault-Tolerant Computing*, pages 150–159, 1998.

[5] K. Tindell, A. Burns, and A. Wellings. "Analysis of Hard Real-Time Communications". *Real-Time Systems*, 9(2), Sept. 1995.

Byzantine Fault Containment in TTP/C

Günther Bauer Hermann Kopetz Wilfried Steiner

Institut für Technische Informatik
Vienna University of Technology
Treitlstr. 3/3/182.1
A-1040 Vienna, AUSTRIA
E-mail: {sc,hk}@vmars.tuwien.ac.at

Abstract

The TTP/C protocol is a communication protocol for safety-critical real-time applications. It is designed to meet both the cost constraints of the automotive industry and the stringent safety constraints of the aeronautics industry. This is achieved by using the static nature of the TTP/C communication pattern to build relatively cheap communication controllers being supervised by guardians that protect correct nodes from faulty ones. The complexity and, thus, the costs of these guardians determine the type of node failures a TTP/C-based network can tolerate. In this paper, we will give a short overview of the TTP/C protocol and discuss its fault hypothesis. We will then introduce a general guardian that enables a TTP/C-based network to tolerate arbitrary node failures.

1. Introduction

The Time-Triggered Architecture (TTA) is a distributed computer architecture for highly dependable real-time systems. The core building block of the TTA is the communication protocol TTP/C. This protocol has been designed to provide non-faulty nodes with consistent data despite the presence of faulty nodes or a faulty interconnection network channel. To achieve consistency the protocol algorithms assume that a fault is either a reception fault or a consistent send fault of some node. Although the protocol uses this rather optimistic failure mode assumption, the TTA can isolate and tolerate a broader class of faults. This is possible by making intensive use of the static knowledge present in a TTP/C-based distributed computer system. This off-line available knowledge allows to build interconnection networks which transform arbitrary failure modes of nodes into failure modes the communication protocol can deal with.

This paper discusses a promising new approach to transform failure modes in TTP/C-based systems utilizing a star topology interconnection network: the central guardian [3, 1]. In a star network architecture all TTP/C nodes can share guardians that are physically located at the star coupler of the network. This setup requires only a single guardian per replicated communication channel rather than a guardian for each node as needed in a bus setup [6]. Thus, the guardian may implement sophisticated algorithms while keeping overall system costs low. In fact, a central guardian may even be designed as smart as to isolate arbitrary node failures.

The remainder of the paper is organized as follows: we start with a short introduction to the TTP/C communication protocol and will present its fault hypothesis. We will then discuss the requirements imposed on a guardian for TTP/C that enables isolation of arbitrary node failures.

2. The TTP/C Communication Protocol

A TTP/C network consists of a set of communicating nodes connected by a replicated interconnection network. A node computer comprises a host computer and a TTP/C communication controller with two bi-directional communication ports. Each of these ports is connected to an independent channel of a dual-channel interconnection network. Via these broadcast channels the nodes communicate using the service of the communication controller.

The TTP/C protocol implements broadcast communication that proceeds according to an *a priori* established time-division multiple access (TDMA) scheme. This TDMA scheme divides time into slots each being statically assigned to a particular node. During its slots the node has exclusive write permission to the interconnection network. The slots are grouped into rounds: in the course of a (TDMA) round every node is granted write permission in exactly one slot.

This work has been supported by the European IST project *Next TTA* under project No IST-2001-32111.

Furthermore, nodes always send in slots having the same relative position within a round; finally, the slots assigned to a particular node have the same length in each round. A distributed fault-tolerant clock synchronization algorithm establishes the global time base needed for the distributed execution of the TDMA scheme.

A cluster cycle comprises several TDMA rounds and multiplexes the slots assigned to a node in succeeding TDMA rounds between different messages produced by the node (this is similar to the TDMA round, which multiplexes the communication channels between several nodes). Every node has knowledge – stored in read-only memory – of the complete communication pattern (and not only of the slots assigned to itself). These data are called message descriptor list (MEDL) and allow nodes to know *a priori* the types of messages being sent or received. Thus, there is no need for transmitting the sender IDs or message IDs explicitly.

TTP/C messages are called frames and the protocol defines three types of frames: normal frames (N-frames) carry user data. Initialization frames (I-frames) carry protocol-specific state information that allows nodes to integrate into an operational cluster. Finally, extended frames (X-frames) contain both user data and protocol state information. The type of a frame to be transmitted in a particular slot of the TDMA round is also stored in the MEDL. In addition – to allow for node integration – frames carry an identifier bit in a frame header.

By periodic examination of frame states the protocol establishes a membership service: if a node receives a correct frame [4] on either of the communication channels, it considers the respective sender correct. A correct receiver will consider a frame correct if it meets all of the following requirements:

- transmission of the frame starts and ends within the temporal boundaries of its TDMA slot
- the signal constituting the frame on the physical layer obeys the line encoding rules
- the received frame passes a CRC check
- sender and receiver agree on the distributed state of the TTP/C protocol (i.e., the C-state)

It does not matter if the sender is in fact correct (as judged by an omniscient observer) or what faulty receivers conclude. If a node receives a correct frame, it assumes that the contents of the frame are authentic and that sender and receiver agree on the distributed state of the communication system, i.e., the controller state (C-state). The C-state consists of the membership, the global time the frame broadcast was started at, and the number of the current TDMA slot. To test C-state agreement when an N-frame (which contains solely user data) is received, the CRC check mentioned above is performed on the frame data concatenated with the local C-state (extended CRC check [4]). If the resulting CRC checksums are identical at sender (i.e., the checksum transmitted with the frame) and receiver, the receiver assumes that it maintains the same C-state as the sender. Alternatively, when an I-frame or an X-frame is received, the C-state data transmitted with the frame are compared to the receiver-local view of the C-state. In any case the membership service of the protocol ensures that a node can only succeed in broadcasting frames if it maintains a correct C-state (i.e., the same C-state as the receivers).

To allow for integration of nodes into an active cluster, some nodes of the cluster periodically broadcast their respective C-state in I-frames or X-frames. Nodes willing to integrate can learn membership, global time, and the actual position within the global communication pattern from the C-state. Thus, the node is enabled to participate in communication after having received an I-frame or an X-frame.

3. TTP/C Protocol Fault Hypothesis

In the following paragraphs we will introduce the fault containment regions [2] of the TTP/C protocol. Further, we will provide definitions of types and frequency of faults that can be withstood by the protocol. Finally, we will define a minimum configuration needed to tolerate these faults.

3.1. Fault Containment

The TTP/C protocol distinguishes between two types of fault containment regions:

- node computers (comprising a host computer and a communication controller part) and
- channels of the interconnection network.

A fault containment regions is supposed to fail as a unit. Distinct fault containment regions will fail statistically independently if the respective faults are covered by the fault hypothesis.

3.2. Node Faults

As for the frequency of node faults, the fault hypothesis of the protocol claims that

1. only one faulty node exists within the duration of a TDMA round
2. a node may become faulty only after any previously faulty node either has shut down or operates correctly again.

With respect to the types of node faults, the TTP/C protocol assumes that

3. a transmission fault is consistent (i.e., if a faulty node broadcasts a frame on a correct channel, all receiving nodes will consistently consider the respective frame faulty or correct)

4. a node does not send data outside its assigned sending slots on both channels of the interconnection network

5. a node will never send a correct frame outside its assigned sending slots

6. a node will never hide its identity when sending frames.

The fault hypothesis does not state anything about faults other than communication faults. Any fault of a node (even a reception fault) will either become manifest by a transmission fault of the affected node or will never be perceived by other nodes of the cluster.

3.3. Network Faults

With respect to the frequency of faults of a channel, the fault hypothesis states that

7. only one channel is faulty during a TDMA slot.

As to the types of interconnection network faults it must be guaranteed that

8. a channel does not spontaneously create correct frames

9. a channel will deliver a frame either within some known maximum delay or never.

3.4. Single Faults & Minimum Configuration

The TTP/C protocol promises to provide its consistent frame delivery and membership service even in the presence of faults provided that at most one component happens to be faulty in a particular slot. To achieve fault-tolerance, however, a minimum configuration must be ensured.

To tolerate a faulty node the minimum configuration in TTP/C requires in general that, in every slot, there exist at least three correct nodes which need to be correct for the whole duration of the slot. In particular, if the cluster operates in synchronous protocol mode, three correct nodes, which must actively participate in clock synchronization and are synchronized to each other, are required in a minimum setup. Further, to allow for integration of a correct node despite a faulty active node, an I-frame must be transmitted every TDMA round and there must be at least one correct node that sends I-frames.

4. The Tasks of the Guardian

The purpose of the guardian is to increase the probability that TTP/C nodes of a cluster will face only faults covered by the fault hypothesis as presented in Section 3. In principle, this is achieved by placing a guardian at the interface(s) of a component and let it control the appearance of the respective component at its interface(s) to other components and, thus, act as a failure mode converter. Consequently the failure modes of the component are – at the interface to other components – replaced by the failure modes of the guardian.

The central guardian discussed in this paper checks (at the operational level of the interface specification [5]) for the expected syntax and the timing at the interface of the nodes it supervises. It is thus able to transform types of faults. At its output interface the guardian will mirror the input received from the attached sender node if this input complies to some specified rules. Otherwise the guardian will exhibit a predefined behavior (that complies to the fault hypothesis).

To guarantee compliance to the types of node faults the guardian needs to transform the following failures of TTP/C communication controllers:

1. SOS failures in the line encoding of frames at the physical layer

2. SOS failures with respect to the timing of frame transmission

3. transmission of any data outside the assigned sending slot (both in synchronized cluster operation and during startup)

4. masquerading of nodes during the startup phase of the protocol.

Additionally, to provide fault isolation to integrating nodes:

5. transmission of invalid (i.e., non-agreed) C-state data.

Transformation of failure modes one and two ensures that transmission faults will be consistent. Supervision of failure mode three will guarantee that a node can never send anything outside its assigned sending slots. Finally, hiding its identity becomes impossible to a node if the guardian checks for cheaters. Thus, all assumptions regarding the types of faults as discussed in Section 3.2 are covered.

4.1. The Central Guardian Approach

Figure 1 provides the (logical) top level architecture of a TTA cluster utilizing a star topology network. The cluster comprises four regular nodes, two dedicated nodes, and two star couplers. The regular nodes are connected to each

of the replicated channels of the (star topology) interconnection network via bi-directional links. Two independent central guardians are located at the center of each communication channel, i.e., at the star coupler. The guardian of a channel controls all the (frame) traffic at the respective channel. To achieve this, the guardian needs to be provided with the TTP/C clock synchronization service and needs to have access to C-state data. A dedicated node consisting of a TTP/C protocol controller provides these services (by providing the central guardian with a regular TTP/C protocol interface, i.e., the CNI). This controller is logically (as depicted in Figure 1) a regular TTP/C controller that does not send any frames and whose existence is thus transparent to other nodes in the cluster. Physically, the controller is located at the star coupler and is part of the guardian itself.

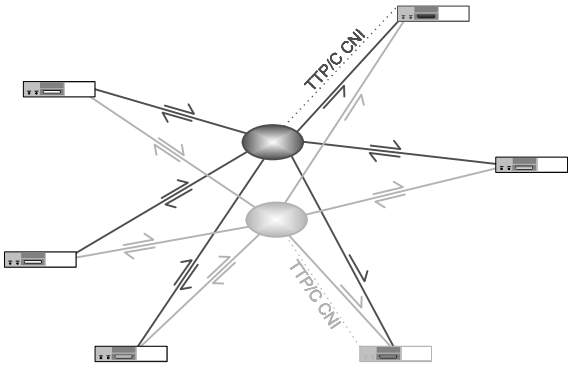


Figure 1. Star Topology Cluster

This approach provides both cost efficiency and a low statistical dependency of node and guardian faults. Cost efficiency is a consequence of needing only two guardians (one per channel) irrespective of the actual number of nodes in the cluster. Because of the strict "one-at-a-time" communication pattern of TDMA-based communication and the fact that a guardian protects receivers from faulty senders, it suffices to have, for all nodes, a single common guardian that is – at a particular point in time – logically assigned to the sender of the respective slot. Write access of a node is prohibited outside its respective sending slot.

The actual value of statistical dependency of node and guardian faults basically depends on the particular implementation. Influencing parameters are the type of physical connection between nodes and the star coupler, independence of power supplies, physical vicinity of the devices, and others. At the logical level nodes and guardians do not have any common mode failure modes.

Further, integrating a central guardian into the star coupler of a star network has the following advantages:

- The algorithms in the guardians can be extended to provide additional monitoring services, such as condition-based maintenance.
- If the guardians reshape the physical signals, the architecture becomes resilient to arbitrary node faults.
- Point-to-point links have better EMI characteristics than a bus and can easily be implemented on fiber optics.

5. Outlook

In this paper we have presented the TTP/C protocol, its fault hypothesis and minimum configuration requirements and the principles of a central guardian for this protocol. The central guardian is a natural yet powerful choice in star network topologies. Because a whole TTP/C cluster needs only two of these central guardians, its design is less constrained by cost arguments than a local guardian needed once (or even twice) for every node. In fact, a central guardian may contain algorithms so sophisticated that arbitrary node failures can be tolerated.

Currently, we are designing the algorithms for a smart central guardian to be applied in a star network topology. This guardian will be able to isolate arbitrary node failures, thus, allowing to waive sophisticated self-checking mechanisms when needing to ensure fail-silence failure semantics. Test series with first prototypes of this central guardian both in VHDL simulation and on an FPGA-based hardware prototype implementation provided promising results.

References

[1] G. Bauer, H. Kopetz, and P. Puschner. Assumption Coverage under Different Failure Modes in the Time-Triggered Architecture. *8th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2001), Antibes Juan-les-pins, France*, pages 333–341, Oct. 2001.

[2] C. Jones, M.-O. Killijian, H. Kopetz, E. Marsden, N. Moffat, M. Paulitsch, D. Powell, B. Randell, A. Romanovsky, and R. Stroud. Revised Version of DSoS Conceptual Model. Project Deliverable for DSoS (Dependable Systems of Systems), Research Report 35/2001, Technische Universität Wien, Institut für Technische Informatik, Treitlstr. 1-3/182-1, 1040 Vienna, Austria, 2001.

[3] H. Kopetz, G. Bauer, and S. Poledna. Tolerating Arbitrary Node Failures in the Time-Triggered Architecture. *SAE 2001 World Congress, March 2001, Detroit, MI, USA*, Mar. 2001.

[4] H. Kopetz. *TTP/C Protocol – Version 0.5*. TTTech Computertechnik AG, Schönbrunner Straße 7, A-1040 Vienna, July 1999. Available at http://www.ttpforum.org.

[5] H. Kopetz. On the Specification of Linking Interfaces in Distributed Real-Time Systems. Unpublished draft, Technische Universität Wien, Institut für Technische Informatik, Treitlstr. 1-3/182-1, 1040 Vienna, Austria, 2002.

[6] C. Temple. Avoiding the Babbling-Idiot Failure in a Time-Triggered Communication System. In *Proceedings of the 28th Annual International Symposium on Fault-Tolerant Computing (FTCS-28)*, pages 218–227, June 1998.

Internet – technologies that are missing

CACH Petr
cach@feec.vutbr.cz
Department of Control and Instrumentation
Brno University of Technology
Božetěchova 2 612 66, Brno,
Czech Republic

FIEDLER Petr
fiedlerp@feec.vutbr.cz
Department of Control and Instrumentation
Brno University of Technology
Božetěchova 2 612 66, Brno
Czech Republic

The industrial communication networks face a serious challenge – how to achieve seamless interoperability of fragmented networks operating on heterogeneous protocols and how to access data from these networks over Internet to fulfill the idea of ubiquitous computing.

Past development of interoperability in the area of industrial automation can be summarized as follows:

1. Devices from different vendors on single network;
2. Unification of meaning of data fields including physical quantities;
3. Creation of more complex networks
    a. Utilization of different media on one network;
    b. Development of inter-networking (protocol translating) gateways

The big question is: "*What is next?*"

1. Introduction

Today we can see that Internet is covering all the Earth. Sooner or later an Internet connection will be available for a reasonable price almost everywhere. The only known limit of present Internet - the lack of IP addresses - will be solved by IP version 6, so the Internet is not declining technology at all. In fact the necessary technologies are developing towards the idea of ubiquitous computing. As the Internet technologies are spreading, there are many attempts to utilize Internet for transmission of automation data too. Almost every vendor or vendor group has developed a method that allows to interconnect their fieldbuses over Internet. However all these solutions share common drawbacks:

- Misunderstanding the philosophy of Internet;
- Attempts to use former (sometimes pretty old) protocols over Internet;
- No attempts to utilize advantages of Internet;
- Only half-opening of the protocols while presenting them as open protocols.

Many people believe that Internet is just a huge network that interconnects many computers. That's wrong. Internet is a network that interconnects networks. Those interconnected networks are called subnets (subnetworks). Every computer that is connected to Internet is primarily connected to a subnet. Even a single computer that connects to Internet using modem or cable becomes a member of some subnet for the duration of the connection. To access the Internet every computer uses an Internet gateway that is present on the subnet. The gateway is the only device directly connected to the outside world – the Internet.

After we accept that Internet interconnects subnets, we can easily find out that the best way to connect arbitrary devices to Internet is to connect them to a network. To a network that can be used as an Internet subnet. So if we want to connect a temperature sensor directly to Internet we have to equip the sensor with communication interface that can connect the sensor to a network which can be connected to Internet. Moreover the sensor has to use Internet compatible protocol.

At present time it seems that the primary goal in Internet based automation is the development of devices that can be connected to Internet - anyhow - and still utilize the old protocols. The proposed idea is that the primary goal should be: *"Let us develop a technology that will enable to connect automation devices to an Internet compatible network, which will form an Internet subnet. Let us develop Internet compatible fiedlbus*

protocol that will fully utilize the advantages of Internet."

To find out what communication technologies have to be developed, it is necessary to define what is Internet compatible network and what is Internet compatible protocol.

2. Internet compatible network

Common for all Internet compatible networks is that there exists a specification defining how to transmit datagrams of Internet Protocol over these networks. All useful data is then encapsulated in the datagrams of IP protocol. So Internet compatible networks are all networks that are able to transmit unaltered datagrams of Internet Protocol (IP protocol). The content of the useful payload, layers below IP protocol and the technology used as physical layer are not important.

3. Internet compatible protocol

If a protocol is Internet compatible it means that the protocol operates above the IP protocol and uses the IP protocol to transport data.

4. Internet compatible technologies

It is obvious that to connect devices to Internet we have to use some Internet compatible technology (network + protocol). You may say: "*O.K that's what we are doing all the time. We have a device that has Ethernet interface and Ethernet is Internet compatible network. The device uses HTTP and FTP protocols that are Internet compatible protocols, so what's all this about?*"

Well, if you have a device that acts as a HTTP or FTP server, then it is O.K. However there is no suitable technology that would enable to connect low cost sensors (or other devices) to Internet. The manufacturers of sensors have following requirements on communication interface:

- low cost;
- low power consumption;
- bus technology (not star technology);
- minimum space requirements.

These days almost all Internet compatible automation devices are equipped with Ethernet interface. The reason is simple - there is no reasonable alternative to Ethernet. However Ethernet is not low power technology, nor it is space efficient and the most used variant 10/100BasedT is based on start topology with active network components (HUBs and Switches).

As an example this paper presents an idea to use CAN 2.0B based networks for transmission of IP datagrams to fulfill the above-mentioned requirements. The CAN is slow (up to 1 Mb/s), however the amounts of data that are being sent from sensors are extremely low. On the other hand the CAN meets the need for low power consumption and low space requirements. The creation of "process area network" based on CAN could bring the Internet to process level while maintaining high reliability and low cost. The CAN/Ethernet gateway can assure real-time processing on CAN, processing of non-real time requirements and many attractive features like encryption and authentication, firewalling, etc.

Another attractive option for some industrial and process applications are wireless communication networks based on Bluetooth, IEEE 802.11 and IEEE 802.16 technologies, that are very fast, flexible and thanks to their wireless nature do not require any cabling thus are suitable for applications that require to interconnect mobile, moving or rotating objects.

5. Application layer protocols for automation

Even with CAN based Internet compatible subnet there is still missing Internet compatible automation protocol. There had been developed many automation protocols in the past, however these protocols had been tailored to specific applications. These "obsolete" protocols are not able to utilize advantages of Internet and most of them are not open. We believe that there is need for new automation protocol, which would fulfill following requirements:

- Fully open technology - the protocol has to be published as RFC document (freely downloadable);
- Utilization of IP addressing scheme (utilization of IPv6 as the IPv4 is obsolete for intended applications);
- Definition of data as physical quantities including units of measurement (as in LonWorks or CANAerospace), which would minimize interoperability issues.

At present time many proprietary protocols for IP networks are under development in various European and foreign countries, e.g. ProfiNet, Ethernet/IP, IDA, Modbus/TCP, etc. Development of protocols performed by competing vendor-group organizations will not lead to solution that could guarantee interoperability among the animus vendor groups. The only way that can lead to development of open, reliable, secure and globally acceptable standard for exchange of sensor/actuator data is unbiased cross-national development based on real needs of the users. The competing vendor groups are biased towards their old protocols and their will to create something new from scratch is quite low. However without globally acceptable standard that would

guarantee global interoperability of automation data the interoperability problems will remain.

However the possibility to start with a new protocol from a scratch provides another attractive possibility. Almost all present automation protocols lack security features, however the need for such features is significant. The advantage of Internet is the wide selection of available security technologies for encryption and authentication. Especially the ability to create encrypted tunnels is very attractive for automation purposes. Encrypted tunnels allow creation of secured wide-area data acquisition and control systems, while keeping computational requirements placed on automation devices very low.

The new application layer protocol should be based on the best ideas selected from all automation protocols. For example the idea of physical quantities based data representation (as in LonWorks and CANAerospace protocols) can eliminate large amount of interoperability issues that are quite common in these days. Protocols designed with interoperability on mind can guarantee that the communicating devices will be able not only to "talk" together (common protocol), but also it is guaranteed that the devices will understand to each other (common interpretation of data structures) as the protocols exactly specify how to store and transmit values of physical quantities, including units of measurement. An apposite demonstration of the significance of the common data interpretation is the unsuccessful NASA mission to Mars with Mars Climate Orbiter in 1999. The official explanation of the crash from NASA was: "*The 'root cause' of the loss of the spacecraft was the failed translation of English units into metric units in a segment of ground-based, navigation-related mission software ...*"

Obviously engineers at NASA and their operation guidelines were not aware of the interoperability issues.

6. Conclusion

The presented paper introduces the key technologies needed for successful penetration of Internet into the area of industrial automation. The era of ubiquitous computing is slowly approaching, however the technology is not ready yet. Missing are low cost, low power IP compatible communication technology, IP compatible application layer protocol with state of the art security features, proven wireless IP compatible technologies and widespread use of IPv6 protocol.

With described technologies the Internet could be used as widely available fieldbus that would allow for applications, which are hard to imagine with present technologies (e.g. global data acquisition systems for weather forecasting, disaster prediction, traffic control, pollution control, remote health status monitoring of persons with medical issues, citizen based remote supervision of environmentally sensitive plants, etc.).

This paper briefly introduces thoughts and ideas that are behind the Expression of Interest (EoI-CZ27) named *Communication and software technologies for embedded, distributed and Internet based automation* submitted in the frame of Integrated Projects of 6th Framework Program.

The paper was inspired by results of research and development that was supported by Ministry of Trade and Industry of the Czech Republic in the frame of project FD-K/104 during co-operation with BDSensors Ltd.

7. References

[1] Industrial Automation Open Networking Alliance, http://www.iaona.com

[2] The Industrial Ethernet Book, http://ethernet.industrial-networking.com

[3] Bradáč Z., Fiedler P., Zezulka F., "Heterogenous interconnection of industrial fieldbuse", Proc. of IWCIT99, Ostrava, 1999

[4] Cach P., Fiedler P., Zezulka F., "Sensor/Actuator web oriented interface", Proceedings of PDS2001 IFAC Workshop, Gliwice, Poland, 2001

[5] Cach, P., Fiedler, P., Zezulka, F., Vrba, R., Švéda, M. "Internet based remote I/O". In Summaries of 16th International Conference on Production Research ICPR-16, Praha, 2001

[5] NASA. "Mars Climate Orbiter failure board report release: 99-134", 1999, available on Internet at address http://mars.jpl.nasa.gov/msp98/news/mco991110.html

[6] Vergeest, J. – Horvath, I.: "A fundamental limit of interoperability". In proceedings of 10th Symposium on Product Data technology, Quality marketing Services, Sandhurst, U.K., 2001

Ethernet interface in application – case study

CACH Petr
cach@feec.vutbr.cz
Department of Control and Instrumentation
Brno University of Technology
Božetěchova 2 612 66, Brno,
Czech Republic

FIEDLER Petr
fiedlerp@feec.vutbr.cz
Department of Control and Instrumentation
Brno University of Technology
Božetěchova 2 612 66, Brno
Czech Republic

The paper describes development of three industrial devices, which utilises Ethernet interface. The first two are designed to allow connection of already existing systems to the LAN. Both systems assume that the connected system is already equipped with serial port and implements some type of communication protocol. The first one crates virtual serial port, so already existing software can be utilised. The second system works as a specialised web server. It implements a universal script language, which allows to send ad receive data through serial intarface and dynamically create web content.

The third device is a datalogger equipped with number of universal analog and digital inputs. The collected data are recorded in the memory of the datalogger and accessible using embedded web and ftp server.

1. Ethernet in automation

Considering the advantages of modern computer communication technologies, especially Ethernet, it is in place to ask how to utilize them in an industrial system (by the industrial system we can understand a control system, PLC, smart controller or measurement instrument). When building a new system, the answer is evident, simply to buy one with already implemented Ethernet interface. However, in case of already installed system (or when the manufacturer doesn't offer an Ethernet variant) it is necessary to utilize some alternative solution.

When looking for such solution it is desirable at first to ask, what effect should the Ethernet bring to the user? Then according to the desired effect it is possible to choose a proper solution. Basically, the expected effect is to get a remote access to the system, either over local intranet or over Internet.

When examining existing industrial systems one can find that the common feature of majority of the systems is presence of some kind of serial interface (RS-232/422/485), regardless of the used communication protocol. Considering this, the obvious way of connecting any already installed industrial system to the local network is to use some kind of Serial/Ethernet converter/gateway. Functionality of such gateway can vary according to the user's needs; from simple bridge to sophisticated embedded web sever.

2. Serial/Ethernet gateway – virtual COM port

The application of the S/E gateway assumes that the connected system is already equipped with a serial communication port and there is some software utilizing the communication link. The S/E gateway guarantees the transfer of data between PC and the remote device over Ethernet.

The simplest way is to use only bridge, which tunnels the serial protocol of the industrial system over intranet or Internet, with dedicated gateways on both sides of the communication link. Such approach is suitable for interconnection of two non-PC devices. However, when connecting a system to PC, this solution has some substantial drawbacks. In fact, it is cumbersome to convert the data from Ethernet back to the serial line, since it is much more simple to equip the computer with a standard network card. The gateway on the PC side is replaced by software, which simulates operations of a standard COM port (see Figure 1.). There is no need to

change the user software; the user only selects the virtual COM port in the configuration of his software. There is no need for dedicated hardware also, so there is no limitation on the number of computers, which can be connected to the system. The only limitation is that there can be only one computer currently connected to the remote system.

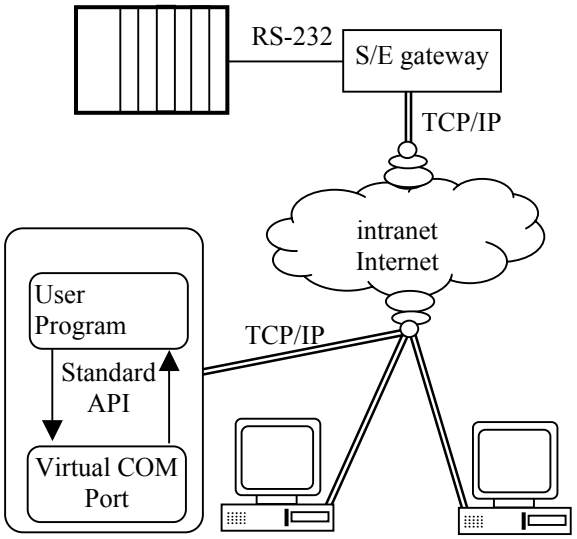


Figure 1. Virtual COM Port

The whole project was separated to two parts. The first part was the software on the PC side. The software is developed for Windows 2000/XP, although it can be easily ported to different version of Windows. It consists of three components – driver, port configurator and gateway configurator. The function of the driver is obvious. It simulates the COM port functions, collects requests from the Windows system and performs them and communicates using TCP/IP protocol with the gateway. The communication protocol, which transfers data between the driver and the gateway, uses both raw socket connection for data transmission (data written to the socket on the PC side are directly written to the serial port of the gateway a vice versa) and Modbus TCP protocol for configuration of the serial port (baud rate, parity, handshake, etc).

The second part, port configurator, allows the user to automatically detect present devices on the local network and install the virtual COM port. It is also possible to enter list of devices which are not on the local subnet, but are located in another network where can't be automatically detected (for the detection of the present gateways is used a broadcast UDP message, which can be received only on the given subnet).

The last part of the software is the gateway configurator. It is used to set IP address and other TCP/IP parameters of the interface. To be able to configure selected gateway, the device has to be on the same subnet as the computer running the configuration program. It can scan local network for the already configured devices (it means they have valid IP address) or configure a new device using it's MAC address only. There is utilised an UDP broadcast message for this purpose.

The hardware of the gateway is based on Ethernet module RCM2200, developed by Rabbit Semiconductor. This module integrates an advanced 8-bit microprocessor Rabbit2000, a standard Ethernet controller Realtek8019AS (10Mbit/s) and sufficient amount of memory. These modules are suitable compromise in price/performance ratio. Connected to the module is a daughter board with a 5V power source, small serial EEPROM to store the actual configuration and the drivers for RS-232 and RS-485, so the interface provides two independent serial ports.

3. Embedded web server

The previous solution is suitable in cases when there is available an already existing software managing the industrial device. The virtual COM port only makes a remote access link to the distant system, either over local network or Internet. However there are situation where this solution isn't suitable, e.g. when there is a need to share data from the industrial process over web.

The first solution, which is available, is to use the previous type of interface together with a PC to collect data, where the PC with installed web server is used to publish the data on the web. However much better idea is "why don't we include the web server already into the gateway?"

The embedded web server has to be flexible to be able to work with different types of industrial systems. The web server has to communicate with the system and dynamically generate html code, which will reflect the actual data. From this reason a simple scripting language is implemented in this device. The script enables the user to implement any serial communication protocol. It has similar syntax to the JavaScript used in the web browsers. The function set is sufficient to work with the serial port, send data to the serial port, receive information, process them and finally put the result in the generated html page. To define dynamic content the web server uses so called "server side includes". The server side include is a special command included directly in the source code of the html page. When a client (e.g. Internet Explorer) requests such page the web server processes the commands and replaces them by a result. The cgi script can be also used as the target url in forms defined in the html page, so the user can pass to the script any parameters.

```
Command in the html code:
Outside temperature is <!-- echo
var="temperature">°C.

Result after server processing:
Outside temperature is 25°C.

Command in the html code:
The valve is <!—exec cmd="Script01.cgi">

Script definition:
       function Script01() {
         if(valve)
           _Output_ = "open.";
         else
           _Output_ = "closed.";
       }

Result after server processing:
The valve is closed.
```

Figure 2. Example of the SSI

Such approach providesgives very efficient and flexible tool. Because the system uses standard http protocol for data transfer it is easy to interface it with any third party system. The used mechanism of data transfer even allows using of modern XML language for data representation and its automatic processing. The system can be easily combined with JavaScript or Java applets so the processing and graphical representation of the data from the connected system is done on the user's computer, thus reducing the load of the embedded web server.

The hardware of this embedded web server is again based on the above-mentioned Ethernet module. Both devices can share tools for configuration (IP address, etc.).

4. Standalone data logger

The third device, the datalogger, was developed as a platform that would enable to collect data from a technological process and allow access to all this information from Intranet or Internet without any additional hardware. It was required that no special software should be needed for access to this data. Only standard software such as a web browser or ftp-client was allowed.

In principle, there are two ways how to achieve this goal. One is to embed all the necessary hardware in a sensor and individually connect each sensor to Internet. Although one can fully utilize all features of the sensor (full configuration and parameter setting over Internet), drawbacks such as higher price and lack of suitable components for such task (size, power consumption, price) prevent realization.

The other way is to create a universal device with number of universal inputs and connect standard sensors to them. Such data concentrator offers higher flexibility, possibility to pre-process collected data directly in the data concentrator, lower price per sensor, etc.

Such data concentrator was realized in close cooperation between authors department and BD Sensors s.r.o. company. The device is equipped with 14 analog inputs (0/4 ... 20mA) and 8 universal digital I/O. There is also a serial port allowing to connect GSM modem. The inputs lines are periodically scanned and the measured values are stored in the memory (1MB flash, this capacity enables to store data from all channels for 1 week with 3 minutes period).

The embedded web server enables access to both actual and past data. The server in the datalogger provides dynamically generated html pages. The content of these pages is user defined and can reflect actual measurements provided by connected sensors (it uses the same technique of server side includes as the previous device). The stored data can be displayed aither as a table or using Java applet as a graph. The web server includes also a special configuration page, which allows easy configuration of analog and binary inputs and data storage. The datalogger implements also a ftp server. It allows the user to download stored data as a file.

The devices implements some additional functions. For example it can be configured to send email when some input crosses predefined limits (a SMS when a GSM module is connected). It can also work as a simple on-off controller.

The hardware of the datalogger utilizes Net+ARM microprocessor manufactured by NET+Silicon company. It is 32-bit microprocessor with well-known ARM core. The main advantage of this processor is integrated 10/100 Ethernet controller module, which serves as the Ethernet Medium Acces Control layer. This module, together with powerful ARM core, gives us enough performance for processing of the massive data flow from Ethernet interface and for processing of TCP/IP and application layer protocols. It allows to use both widespread Ethernet standards, 10BaseT and 100BaseT.

5. Conclusion

During development of the described systems it has been found that Ethernet with TCP/IP protocol suite is a

good choice as an automation networking technology that allows easy connection to Internet. The system shows several advantages of Internet technologies in the industrial segment. The TCP/IP protocol allows to use several communication protocol to exchange data, independently on each other.

All three systems were developed on authors department. The first two in cooperation with GMC s. r. o. company, the last one in cooperation with BD Sensors s. r. o. company. One of the effects of this effort was to collect practical experience with Ethernet and it's application in industrial communication. None of the presented device is designed to meet tight real time requirements. The main area of application is remote monitoring and data acquisition.

Acknowledgements

The research has been supported by GMC Ltd., the Faculty of Electrical Engineering and Communication of the Brno University of Technology, Czech Ministry for Education - Research Intention JC MSM 262200012 and Czech Ministry of Trade and Industry of the Czech Republic in the frame of project FD-K/104 during co-operation with BDSensors Ltd.

References

[1] Industrial Automation Open Networking Alliance, http://www.iaona.com
[2] The Industrial Ethernet Book, http://ethernet.industrial-networking.com
[3] RabbitSemiconductor, http://www.rabbitsemiconductor.com
[4] Zezulka F., Fiedler P., „Ethernet v průmyslové automatizaci", Automa n. 7, p. 35-38, ISSN 1210-9592, 2000
[5] Bradáč Z., Fiedler P., Zezulka F., "Heterogenous interconnection of industrial fieldbuse", Proc. of IWCIT99, Ostrava, 1999
[6] Cach P., Fiedler P., Zezulka F., "Sensor/Actuator web oriented interface", Proceedings of PDS2001 IFAC Workshop, Gliwice, Poland, 2001
[7] Cach, P., Fiedler, P., Zezulka, F., Vrba, R., Švéda, M. "Internet based remote I/O". In Summaries of 16th International Conference on Production Research ICPR-16, Praha, 2001

Real-Time with Ethernet

R. Messerschmidt
Otto-v.-Guericke University
Institute for Ergonomics, Manufacturing Systems and Automation
Center Distributed Systems
Universitätsplatz 2, D-39106 Magdeburg, Germany
ralf.messerschmidt@mb.uni-magdeburg.de

Abstract

After a general real-time consideration and classification the opportunities of Ethernet concerning real-time will be discussed, especially the IAONA Real-Time classes will be viewed and achievements of Ethernet components and features are considered in general.

Real-time types

Although the term real-time is often used in the field of Automation and Communication, an academic definition of real-time or even a numeric specifications for real-time behavior can not be given in general but only in close context with a special application of a distributed control. Is a system in all circumstances able to react to all occurring events correctly and timely then it is real-time capable [1]. If a communication system meets all the timely requirements of a certain application it is – related to this application- real-time capable.

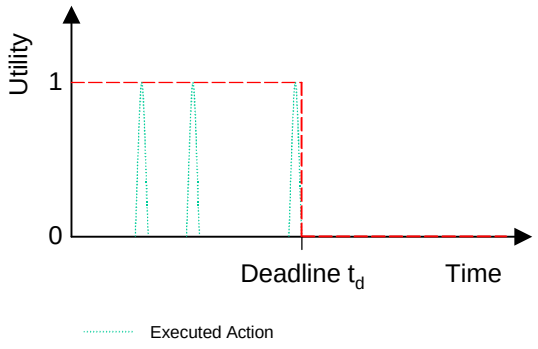


Figure 1: Time/Utility function with an execution deadline (timeliness)

This might mean that an action has to be completed until a maximum time line – the deadline (see Figure 1). This behavior can be described with timeliness.
But often this means too, that actions have to happen on a certain pre given moment (see Figure 2). This behavior can be described with simultaneousness [2].
According to Douglas Jensen´s Time/Utility Function Model [3] of real-time the first case means that the utility of completing an action is 1 until the deadline. The other case means the utility is only 1 in a small window of time. Time/Utility Functions are a more general formulation of time constraints. They express the utility of executing (and completing) an action as a function of when the action is executed (and completed).

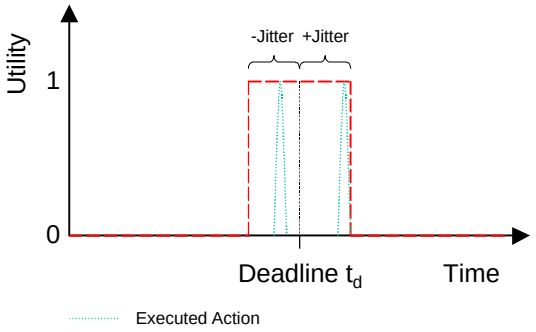


Figure 2: Time/Utility function with execution window (simultaneousness)

The utility values express the relative importance of an action.

Real-time types and Ethernet

In the first case of above described time constraints Ethernet TCP or UDP/IP is an appropriate protocol.
The second above described case can generally not be guaranteed with Ethernet TCP or UDP/IP, variations in the transmission duration (so called jitter) can be

caused by unpredictable delays in buffer queues that are in general too large for those applications. However, this can be overcome by combining guaranteed maximum transmission times (worst case studies) and an appropriate exact time synchronization mechanism. Such synchronization algorithms that will provide a synchronization exactness far beyond one millisecond currently are developed by the IEEE working group 1588.
IEEE 1588 synchronization will provide a technology to synchronize the internal clocks of distributed controls. So commands sent between nodes can be equipped with a time stamp of the desired execution time, thus actions can be executed simultaneously or coordinated on a common system time base. So not the communication is synchronized in general (even if this could also be realized by this technology) but the execution time directly is synchronized.
Synchronized communication is usually realized by time slot technologies, where the right to communicate is allocated cyclic to each node (e.g. by a master or a round passed token) so that nodes of a net can never communicate at the same time and the time difference between sending a command or data by a station and the receipt (and so also the execution) by an other station can be calculated or is pre given. So the nodes work synchronous without a shared common time base.

IAONA Real-Time Classes

The Industrial Automation Open Network Alliance IAONA pursues the aim of establishing Ethernet as the standard application in every industrial environment at an international level. Sense of this is to realize a general, interfaceless communication through all levels of an enterprise.
Concerning Ethernet TCP/IP the IAONA Joint Technical Working Group Hard Real-Time has defined 4 real-time classes that define real-time demands including the aspect of development needs.

Class 1 covers properties that are still available with standard Ethernet products. Class 2 covers products conform to today's standards that are optimized for real-time demands. Class 3 are products with new added functions realized in software combined with standard hardware (of course also this class is down compatible to the standards, but the new added functions can only be used by devices which are prepared for these added functions). Class 4 products have the added new functionality of class 3 but realized in hardware.

Stack optimization

On the shelf stacks today are optimized for TCP not for UDP. So according to IAONA Class 2 an optimized UDP/IP stack, that mostly are a component of the applied real-time operating system, would improve the time behavior for most industrial Ethernet products, since real-time data are usually transmitted using UDP.
Investigations of typical operating systems showed that stacks, as used today, have relatively high throughput times and the fluctuating of throughput times are around factor five. Transmission times on the wire are comparably short on Ethernet so most potential for improvement is in optimizing the processing time in stacks. So for instance a "zero-copy-stack" copies new data only once in a buffer and all the protocol layers of the Ethernet TCP/IP stack access the data at that buffer and not as usual where the data are copied in different buffers for each protocol layer.

Switches

Most industrial Ethernet systems require a switched network with components working with fullduplex to prevent collisions that otherwise – in the case of a shared medium – can occur. In a shared medium end devices must compete for access to the medium. The well know technology for that in Ethernet is CSMA/CD. Switching technology combined with fullduplex avoid the contention for access to the transmission medium, because each end device has one link to the switch for transmitting data and one link for receiving data.
In opposite to an hub - the traditional device for connecting devices over Ethernet - a switch does not mirror data coming in on one port to the outputs of all other ports. A switch knows the addresses of the devices connected on its ports and delivers data only to the addressed target device. Thus a switch is able to provide several connections at the same time. But what happens if , at the same time, data packets from several ports are addressed to the same destination? In that case the data packets are stored in the output queue of the port connected to the target device and will be transmitted to the target device one by one.
Also it has to be considered that he time to transmit packets depends on the packet size so the larger differences in packet sizes the larger the differences in transmission times.

In addition to control data, which requires real-time communication capability, additional data with different load profiles and characteristics must use the network. For example, visualization data, software

updates, e-mail traffic, office applications, and Internet data traffic. For this reason the network must be meticulously designed, to include segmenting those parts of the network where real-time behavior is necessary.
The terminal devices that require real-time behavior should be linked over as few switches as possible. Inevitably, the more switches between two terminal devices, the higher the "worst case" throughput and queue time. With backbones or other instances where there are no factors limiting real-time performance, the individual segments are commonly connected in a ring structure.
In addition, the interface between a real-time segment and the rest of the network must be precisely controlled. Since the data traffic from the general network can adopt any load profile, it must be monitored and restricted when entering a real-time segment. To prevent the real-time segment being overloaded, the amount of data traffic entering this segment must be limited. An effective way to achieve this is to configure the inter-segment link to 10Mbit/s, while all devices on the real-time segment communicate at 100Mbit/s. Further segmentation, as well as access control, can be accomplished by the use of routers and firewalls.

A recent innovation in Ethernet, standardized by the 802.1p working group, is a prioritization mechanism. An additional field, known as a tag, is added to the Ethernet frame. The tag contains information about the priority of the data.
Some Ethernet switches already support this function, Each transmission port has separate queues for the supported priority levels. Data packets in a higher priority queue are always transmitted before those in a lower priority queue.

In systems with strict time constraints the communication rise is usually known so worst case considerations can be done.
By that way systems with the above described execution deadline can be realized.

Ethernet with time slot technology

There are also technologies that provide an already described time slot technology over Ethernet. Since in each time slot only one device can send data there is no need for using switches to prevent from collisions. The use of hubs even improve the time behavior of those systems since hubs are faster than switches, because they do not process the data of the packets and so need less time than switches that have to process incoming data packets to find out its destinations.

As already mentioned above with this technology applications with simultaneousness action execution demands can be realized.

References

[1] F. Furrer: Ethernet-TCP/IP für die Industrieautomation: Grundlagen und Praxis. Huthig Verlag, Heidelberg, 2. Auflage, 2000.

[2] G. Gruhler: Feldbusse und Gerätekommunikationssysteme. Steinbeis- Transferzentrum Automatisierung (STA), Reutlingen, 5. neu gestaltete und erweiterte Auflage, 2000.

[3] E. D. Jensen: Overview of Fundamental Real-Time Concepts and Terms, 2002.

Utilization of Modern Switching Technology in EtherNet/IP™ Networks

Anatoly Moldovansky
Rockwell Automation
1 Allen-Bradley Drive
Cleveland, Ohio 44124 USA
amoldovansky@ra.rockwell.com

Abstract

EtherNet/IP networks are widely used in industrial environments and time-critical applications. In this paper, we characterize traffic generated in a typical EtherNet/IP network and compare it with office network traffic. We provide recommendations regarding features of network switching and routing devices, which, when properly utilized, will help to achieve required performance of EtherNet/IP sub-nets and allow for their successful integration into a plant network. We also provide a list of issues that have been uncovered during these studies.

1. Introduction

Ethernet™ networks have been successfully used on the factory floor for the past 15 years, mainly in non time-critical applications. Evolution of the Ethernet[1] technology from a 10Mbps, half-duplex, bus/tree topology into a 100Mbps and 1Gbps, full duplex, switch/router based hierarchical star topology has created an opportunity for utilizing Ethernet in industrial networks supporting time-critical applications.

Ethernet/Industrial Protocol (EtherNet/IP) is a communication system suitable for use in industrial environments and time-critical applications [1]. It utilizes standard Ethernet and TCP/IP technologies and an open Application layer protocol called Control and Information Protocol (CIP). CIP is also used in ControlNet™ and DeviceNet™ networks. In EtherNet/IP networks, exchange of time-critical data is based on the producer/consumer model where a transmitting device (host or end-node) produces data on the network and many receiving devices can consume the data simultaneously. Implementation of the producer/consumer data exchange is based on the IP (Internet Protocol) multicast service mapped over the Ethernet multicast service.

EtherNet/IP supported functions include:

- Time-Critical data exchange
- Human-Machine Interface (HMI)
- Device configuration and programming
- Remote access to web pages embedded in EtherNet/IP devices
- Device and network diagnostics

2. EtherNet/IP Traffic Profile

In order to identify features of the EtherNet/IP network infrastructure helping to provide required performance and connectivity, it is necessary to characterize its traffic. Within the scope of this paper, EtherNet/IP network infrastructure is defined as a hierarchical interconnection of Layer 2 and Layer 3 Ethernet switches.

Traffic generated during programming, configuration, and diagnostics of EtherNet/IP devices as well as during exchange of non time-critical is normally low-rate traffic that, obviously, has insignificant impact on network performance. Although it contains all three major types, broadcast, unicast, and multicast, this traffic does not require engagement of any special features in the EtherNet/IP network infrastructure.

Broadcast and multicast traffic typically consists of IP packets supporting ARP, BOOTP, DHCP, DNS, SNMP, IGMP and other protocols of this type. Unicast traffic consists of TCP/IP packets.

Traffic generated during time-critical data exchange consists, predominately, of UDP/IP unicast and multicast packets.

Examples include:

- Input/Output (I/O) data and status produced by a remote I/O device for consumption by one or more programmable controllers
- Data produced by a programmable controller for consumption by one or more programmable controllers

[1] More accurately, IEEE Std 802.1™ and IEEE Std 802.3™ technologies.

While EtherNet/IP supports change-of-state reporting, in a typical control system data exchange is predominately cyclic. The time-critical traffic is normally generated at rate of tens of thousands of packets per second, depending on number and type of Ethernet/IP devices and the application. Some EtherNet/IP devices are, for example, capable of generating up to 5,000 packets per seconds. Normally, this traffic is evenly divided between UDP/IP unicast and multicast packets. Packet length is typically less than 100 bytes.

While handling of the UDP/IP unicast traffic does not require engagement of any special features in the EtherNet/IP network infrastructure, handling of the UDP/IP, or IP, multicast traffic does require such engagement.

As it has been already mentioned, IP multicast traffic generated in an EtherNet/IP network is a high-rate, short-packet traffic generated on a continuous basis. For this reason, EtherNet/IP networks differ considerably from typical office networks, where IP multicast traffic is generated sporadically and with much lower packet rates. A growing exception to this traffic profile may be in the area of multimedia audio and video conferencing applications.

An Ethernet Layer 2 switch normally retransmits each received IP multicast, broadcast or unknown unicast packet to all ports. In the example shown in Figure 1, IP multicast traffic produced by remote I/O device I/O_{11} for consumption by Controller 1 will be sent to all devices connected to the switch.

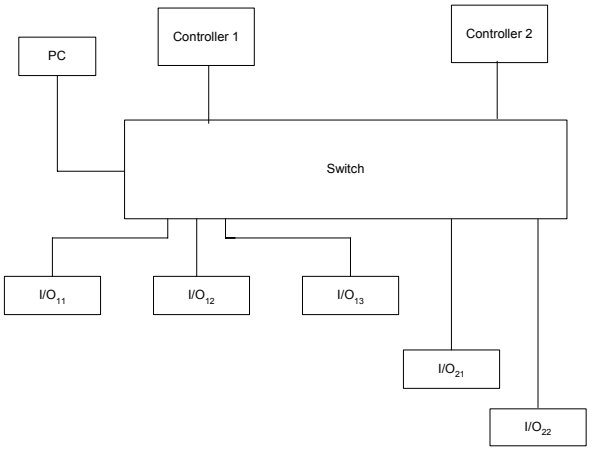


Figure1. Isolated network configured as a single VLAN

Utilization of device resources for filtering this unwanted high-rate traffic can significantly impact device and, consequently, control system performance.

When an EtherNet/IP sub-net is connected to a plant network and propagation of multicast packets through this network is not blocked, it may cause a multicast storm or a flood that will degrade the plant network performance. In an office network, a multicast flood is a temporary event that can be suppressed or controlled. In a plant network with EtherNet/IP sub-nets, the flood of multicast packets is a permanent phenomenon.

Modern Ethernet switches offer a variety of features helping to suppress, block, and route the IP multicast traffic, thus improving network performance, stability, and providing a higher level quality of service. However, not all of these features are effective in dealing with the IP multicast traffic generated in EtherNet/IP sub-nets.

3. Recommendations

In order to optimize network performance, design of the EtherNet/IP infrastructure should be based on the following objectives:

3.1 Minimize device load due to unwanted IP multicast traffic

Depending on sub-net configuration and required device connectivity, this objective can be achieved using Ethernet switches supporting virtual LANs (VLANs) or IP multicast routing.

If a switch is shared between for example, two isolated EtherNet/IP networks, then each network can be configured as a separate VLAN as it is shown in Figure 2. Here, ports 1, 3, 4, 5, and 6 belong to VLAN 1. Ports 2, 7, and 8 belong to VLAN 2. Since IP multicast packets are flooded only to devices inside each VLAN, devices will be less loaded than in the configuration shown in Figure 1.

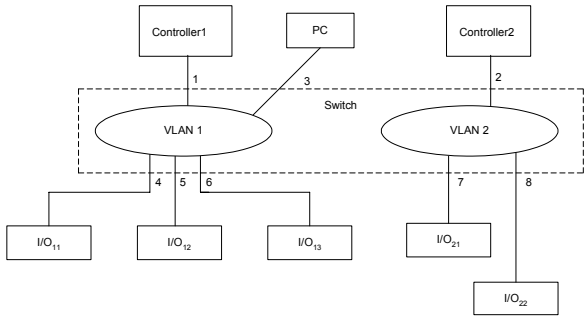


Figure 2. Isolated network configured as multiple VLANs

If EtherNet/IP devices need to share time-critical data, then they have to be connected to the same sub-net. Figure 3 depicts an example of an EtherNet/IP sub-net within a two-layer switch hierarchy. The sub-net is configured at the Layer 3 switch and consists of devices connected to three Layer 2 switches. Support of IGMP

snooping in Layer 2 switches will eliminate device load due to unwanted IP multicast traffic generated in the sub-net. For example, IP multicast packets produced by controller B will be routed only to controllers A, C, and D.

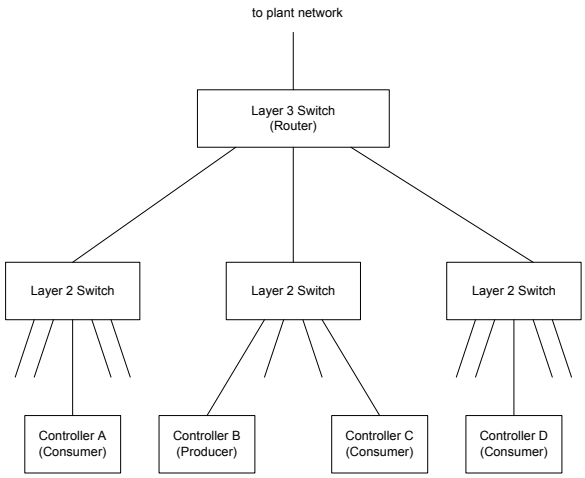


Figure 3. IP Multicast Routing Example

3.2 Minimize switch load due to unwanted IP multicast traffic

Support of IGMP snooping in Layer 2 switches in the configuration shown in Figure 3 will also eliminate switch load with the unwanted IP multicast traffic generated inside the sub-net. If, for example, IP multicast packets produced by Controller B are addressed only to Controller C and IGMP snooping is not used or not supported, then these packets will propagate through all ports of all three Layer 2 switches creating additional load on switches and end-devices.

3.3 Minimize network load due to unwanted incoming IP multicast traffic

The Layer 3 switch in Figure 3 should be configured to block the IP multicast traffic (for instance, stream video) coming from the plant network.

3.4 Block IP multicast traffic generated within the EtherNet/IP sub-net from propagation into the plant network.

This can be achieved utilizing the switch hierarchy shown in Figure 3.

3.5 Optimize switch performance

This can be achieved utilizing switches supporting the IEEE 802.1p priority queuing. In this case, more switch bandwidth can be allocated for time-critical traffic.

4. Issues

The following issues have been identified during performance and interoperability tests of EtherNet/IP networks:

- Lack of interoperability between products from different network switch vendors.
- Inconsistency of IP multicast control features (what they do and how they work) between network switch vendors and in some cases even between different classes of products produced by the same vendor.
- Lack of IP multicast control, support of the IEEE 802.3 spanning tree protocol and other appropriate features in some low-end switches, which considerably limits their use in non-isolated EtherNet/IP networks.
- Lack of industrial high-end Layer 2 and Layer 3 switches.

5. References

[1] EtherNet/IP Specification, available on www.odva.org.

6. Trademarks

IDA - Ethernet based Realtime LAN for Automation Applications

Dipl.-Ing. (FH) Martin Buchwitz, IDA-Group, www.ida-group.org, mbuchwitz@jetter.de

Abstract

IT- and automation technologies come more and more together. Ethernet TCP/IP could seen as one of the basic technology in this context. The reason is, that communication is one of the most important area in the automation market. Devices are getting smaller and they include there own intelligence. Now its the discovery to create a network that could be handled by the application engineers and the service engineers. In the fieldbus world this is a real problem, because of the hierarchical automation pyramid.
Distributed intelligence is the way to fold up this hierarchical pyramid. Distributed intelligence means a free communication between all automation devices. Every information is in Real-Time available for every other device. So the application engineer must not write any line of program code for the communication between the different devices. All this will be done by the system based on distributed intelligence. Interface for Distributed Automation (IDA) is a standard for distributed intelligence in automation. One of the main part is the Real-Time communication over Ethernet TCP/IP.

Motivation

The traditional programmable logic controller known as PLC is a combination of a central processing unit with centralized peripheral modules with a less or more simple structure.

The arrival of fieldbusses has led to migration of these modules to the sensors and actors installed in the field.

The driving force behind this development was to reduce planning and installation costs of extensive parallel wiring with its costly interface levels.

Technological and functional modules have shaped up due to production's growing demands in terms of quality and quantity with ever-increasing complexity of manufacturing processes.
Robots, bonding and welding controllers, closed-loop process controls as well as drive control systems have turned into intelligent peripheral modules linked with the fieldbus.
These modules are able to execute independent operations autonomously. However, their operation needs to be coordinated within the PLC program.

Due to the growing interlinkage of control systems and, at the same time, the wish to have transparent production processes regarding flexibility, quality data and plant diagnostics, the classical PLC, which was originally autonomous, became the central point of networked plant structures.

This structure symbolizes the principal problem of current automation solutions.
The central position of the PLC requires that in the control program not only operating and control sequences of machines or plants have to be programmed, but also communication with lower-level functional and technological modules, as well as communication with interlinked control systems and connection to the plant management level.
In complex automation structures, the proportion of these administrative tasks in no time can exceed the proportion of the machine control itself which results, for the total service life of a plant, in higher and higher expenses of engineering of more and more complex and confusing application programs.

Drastic reduction in engineering costs and at the same time quality's improvement in the field of application software are top priority in the next developing cycle in automation technology.
In this respect, plans for modular machines are being discussed with ever-increasing intensity. The latter are planned to enable as a modular system to meet individual customers' demands effectively and immediately by configuring standardized and coordinated modules and components.
Components that are ready for use can be integrated into the modular system. This approach facilitates their reuse either in the same or in other applications.

IDA'S basic element: the automation module

The basic element of the module based automation system defined by IDA is the automation module. It is an application-neutral top-quality system module representing a complete function which can be used to solve one or several automation problems without requiring any modification.
Plant engineering using IDA's module based automation system enables already completed automation solutions to be reused since all functions are enclosed in application-neutral automation modules. Therefore, they can be used independently of any specific context.
Since automation modules not only include pure automation technology, it is possible to commission units and sections of machines irrespective of the order, and, thus, significantly reduce throughput time of the complete order thanks to parallel fabrication and prefabrication of units.

Moreover, re-use of an automation module facilitates the import of the entire know-how with regard to this module tested in various application areas into a new automation solution, thus, improving quality and reliability from the beginning. This approach increases maintainability and reduces downtime of a machine or plant at any stage of its lifetime.

Transparent Communication

The IDA communication model is based on an integrated approach containing the following: modeling of communication aspects and network view of the functionality.
IDA device communication is based on existing Ethernet communication standards and protocols, such as IP, UDP, TCP, HTTP, FTP, SNMP, DHCP,NTP and SMTP. The IDA communication system provides realtime as well as non-realtime communication services.
On the whole, non-realtime services are based on the above mentioned Ethernet communication protocols.
Realtime capable communication services (e.g. data distribution, RMI, event notification) use the RTPS (Real-Time Publish/Subscribe)protocol which is based on the UDP protocol. To configure and execute realtime capable communication services, IDA specifies an object-oriented model which builds up a hierarchy of communication objects accessible through IDA-API(Application Programm Interface).
Moreover, IDA–API provides specific support for applications important to safety.

Real-time Communication Services

The IDA real-time communication is based on the use of the RTPS-Protocol (Real-Time Publish/Subscribe). The RTPS protocol and the middleware are built on top of the UDP protocol.
Real-time services generally have the highest priority of all IDA communication services. Depending on the requirements of the application, real-time communication relationships and the associated network traffic may be

- preconfigured or dynamic,
- cyclic or on-demand,
- best effort or reliable,
- point-to-point or group-oriented,

Fortunately the offer can be reduced to four types of API services covering the reasonable combinations.

Data Distribution Services are based on a publish / subscribe mechanism provided by the middleware. The term Data Distribution refers to the fact that in each issue the value of an IDA type-defined application data object is published according to the following criteria:

- The application is interested in using the most recent data values only.
- Data transfer is frequent and usually cyclic. Loss of an issue will therefore be mended in the next cycle.
- Bandwidth use is to be minimized.
- Data transfer must be fast.

Event Notification Services are based on the reliable publish / subscribe mechanism provided by the middleware. The term reliable refers to the following criteria:

- The application is interested in transferring each change in the related data value.
- Data transfer is usually not cyclic. Loss of issues is not permitted.
- In-sequence delivery of issues is required.
- Queuing of issues is required to be able to handle bursts and to allow for re-sending of lost issues.
- Bandwidth use is less important than guaranteed delivery.
- Data transfer must be acknowledged on the RTPS level.
- Data size may be nil (i.e. if the event is just used for triggering).

Remote Method Invocation bases on a reliable client/server mechanism provided by the middleware. The term client/server refers to the following criteria:

- Inherently, the relationship is based on a transaction-oriented point-to-point mechanism.

- The application is interested in having a guaranteed transaction on the network and also a guaranteed response on the application layer.
- Data transfer is usually not cyclic. Loss of issues is not permitted.
- Data transfer must be acknowledged on the RTPS level.
- Queuing of issues is required to be able to handle bursts and to allow for re-sending of lost issues.
- Delivery of requests and responses should be in-sequence.
- Bandwidth use is less important than guaranteed delivery.
- IDA communication management objects are employed which are capable of
- establishing connections dynamically at runtime
- evaluating access paths and resolving them into references to IDA Method objects
- performing execution thread instantiation and access control

These tasks are performed by the IDA Method Client and the IDA Method Server objects.

On-demand Data Exchange Service at runtime is based on the invocation of specific methods which are IDA standard. The Get method allows to read a single attribute as well as the whole data set belonging to an IDA Data object. The term on-demand refers to the fact that most parameters are set up dynamically at runtime depending on the requirements of the involved application components at a specific point in time. As a consequence, additional IDA communication management objects must be employed which are capable of

- establishing connections dynamically at runtime
- evaluating access paths and resolving them into references to IDA Data objects
- performing data access control

The IDA Method Server is optimized for the resolution of Get and Set method calls.

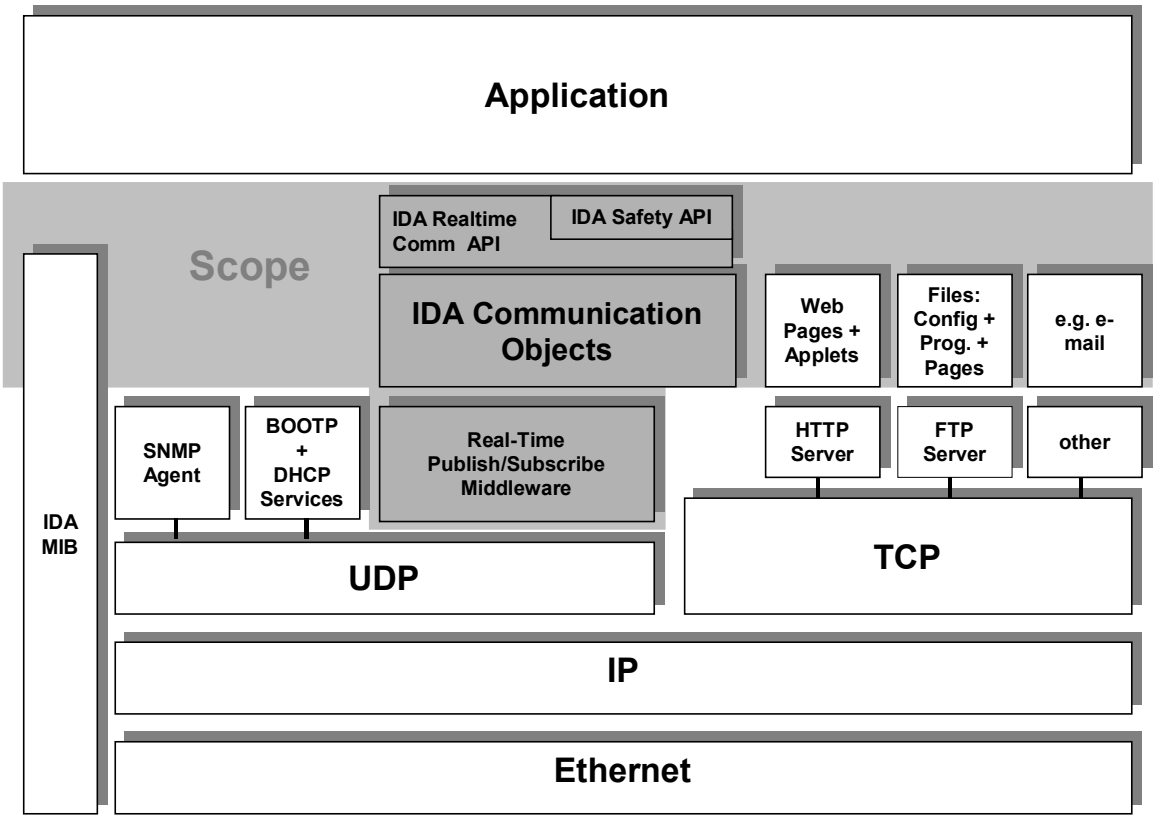


Figure 1: IDA Communication Architecture

Fuzzy Traffic Smoothing: Another Step towards Statistical Real-Time Communication over Ethernet Networks

R. Caponetto[+], L. Lo Bello*, O. Mirabella*

*Dipartimento di Ingegneria Informatica e delle Telecomunicazioni
+Dipartimento di Ingegneria Elettrica Elettronica e Sistemistica
Università di Catania, ITALY
riccardo.caponetto@dees.unict.it, {llobello,omirabel}@diit.unict.it

Abstract

The paper presents an improvement on existing dynamic traffic smoothing techniques in two respects. Firstly, here the input parameters for the smoother are both the overall throughput and the number of collisions observed over an interval. Together, these two parameters represent a more complete indicator of the actual network workload. Secondly, here the smoothing action is dynamically gauged according to the actual workload by using a *fuzzy controller*.

Experimental results in a real environment, comprising 11 workstations running the Linux O.S. and connected via a 10BASE-T Ethernet, are presented, together with a performance comparison with a dynamic smoother in the literature.

1. Introduction

The main obstacle to using Ethernet in real-time communication is that, due to the CSMA/CD access protocol, Ethernet cannot provide connected stations with deterministic channel access times and therefore guarantee that data delivery deadlines will be met. As Ethernet technology today offers a number of appealing features, which suggest adopting it even in time-constrained environments, recently a lot of research addressed the problem of enforcing a predictable behaviour on Ethernet networks.

To support soft real-time applications, which do not require determinism and can accept a *statistical* bound on packet delivery time Shared Ethernet can be adopted, provided that a suitable mechanism to statistically guarantee deadline meeting to real-time packets is implemented. Here we deal with a mechanism, called *traffic smoothing*, which was introduced in [1] and is based on the definition of statistical real-time channels running on an Ethernet. As is known, in an Ethernet, the delay a packet undergoes depends on the number of attempts to transmit before transmission is successfully achieved. In [1] it is demonstrated that a sufficient condition to guarantee with a pre-fixed probability that a packet will gain access to the channel by a pre-established time is that the total rate of new packets generated by the stations remain below a threshold called the *network-wide input limit*. As the Ethernet MAC protocol is totally distributed, a single station is not aware of the current packet arrival rate for the whole network.

Thus, in order to maintain the *network-wide input limit,* each station is assigned a *station input limit* calculated according to the packets' deadlines and the tolerable packet-loss ratio. Each station regulates the *packet stream* arriving from the Application layer in such a way as to keep the packet arrival rate at its MAC sub-layer below the *station input limit* it has been assigned. *Traffic smoothing* only acts on non-real-time (NRT) packets to smooth traffic bursts, as when packets arrive in bursts they are more likely to collide. Traffic smoothing is realised locally in each Ethernet station by a software layer called a *traffic smoother*, inserted between the TCP/IP and the Data Link layer (Fig.1), which buffers any NRT packets arriving in a burst and sends them in such a way to keep their arrival rate at the MAC layer below the *station input limit*.

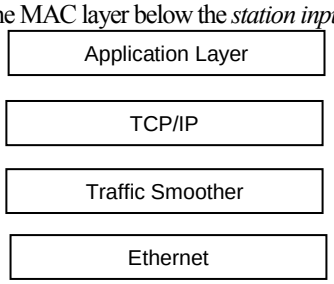


Fig. 1. Software architecture of traffic smoothing

The traffic smoother is implemented by using a leaky bucket-based algorithm [6], where a *credit bucket depth* (CBD), which indicates the capacity of the credit bucket, and a *refresh period* (RP) are defined. Every RP seconds, up to CBD credits are replenished. The CBD/RP ratio is the *station input limit* and determines the average throughput available for a station. By varying the values of RP or CBD, it is possible to control the bursty nature of a flow of packets and thus of the traffic generated by the single stations. When a NRT packet arrives from the IP layer, if there is at least one credit in the bucket, the traffic smoother sends it to the Ethernet Network Interface Card (NIC) and removes a number of credits equal to the size of the packet in bytes. Otherwise, the packet is not transmitted until the next replenishment. A real-time (RT) packet is not affected by smoothing, but its transmission does consume credits. This means that if there is both RT and NRT traffic in a station, the latter is transmitted using any credits that are left over after the transmission of the RT traffic for that station. In *static* traffic

smoothing [1], the *station input limit* is assigned to each station once and for all in such a way that, even in the worst case, each station can be provided with a statistical guarantee on the timely delivery of its packets. This solution has the drawback of entailing a considerable waste of bandwidth, as the *station input limit* is assigned to each station irrespective of the actual load currently on the network (which can even be significantly below the *network-wide input limit*, as not all stations are necessarily transmitting at any one time). Also, scalability problems may arise when the number of stations is high. To provide more scalability and better bandwidth exploitation, in *dynamic* traffic smoothing the [2][3] the *station input limit* is dynamically adapted according to the network workload, thus the available bandwidth is shared only among the stations which really need to transmit. In order to evaluate the network load for dynamic smoothing purposes, different approaches have been proposed in the literature [2][3]. In [2] a dynamic smoother based on throughput control is dealt with, which adapts the refresh period RP to the network throughput measured over a given time interval, while keeping the CBD value fixed. On the other hand, the dynamic smoother described in [3] applies the *harmonic-increase and multiplicative-decrease* (HIMD) algorithm to react to the detection of a *single* collision over an interval. According to the HIMD adaptation, when a packet collision is detected, the RP is increased by the minimum between twice its current value and a given RP_{max} value, while in the absence of collisions the RP is periodically decreased (with period τ) by a constant Δ down to a value of RP_{min}.

Here we propose an approach which extends the previous work outlined above in two respects. Firstly, we use both the total throughput and the number of collisions as input parameters for the smoother. Together, these two parameters represent a more complete indicator of the actual workload. For example, one criticism that can be made of the approach proposed in [3], which reacts to the detection of a single collision over an interval of length α, is that the occurrence of a single collision is not necessarily due to network congestion, but may derive from a temporal coincidence between the transmission requirements of two or more stations when the load on the network is not particularly high. For this reason, regulation of the traffic generation rate should also take total throughput into account. According to the throughput value, in fact, doubling the RP may be too drastic or excessively penalising for NRT traffic.

Another significant improvement introduced here, as compared with the approaches in [2] and [3], lies in the fact that RP regulation is not statically fixed, but is dynamically varied and gauged according to the actual workload by using a *fuzzy controller*. The motivation for choosing a fuzzy controller is that the system considered here, due to its non-linear and quite complex behaviour, is difficult to control using traditional controllers, but is highly suitable for control that is capable of integrating the heuristic knowledge acquired in the field by experts. The choice of a fuzzy approach allows us to embed the knowledge of an expert in the controller. Thus, unlike the approaches in [2][3], where RP regulation is based on fixed variations (doubling the RP or decreasing it by a constant Δ), the action here varies and is differentiated according to the values simultaneously taken by the *two* inputs.

2. Fuzzy smoothing

Fig. 2 gives a detailed block scheme of the fuzzy controller.

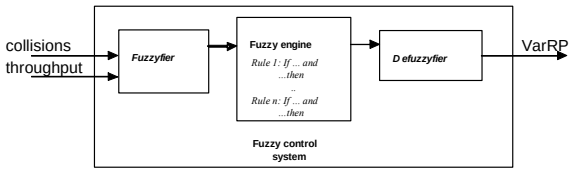


Fig. 2. The fuzzy controller

Our fuzzy controller has two inputs, i.e. the number of collisions and the overall throughput observed in a reference interval, and a single output, i.e. the quantity by which to vary the *refresh period* according to the input values, here called *VarRP*. If RP_{old} is the current *refresh period*, the new value RP_{new} is:

$$RP_{new} = RP_{old} + VarRP \quad (1)$$

Three *membership functions* were defined for each input, corresponding respectively to the values (*Low, Med, High*). This number was chosen heuristically as a tradeoff between representing all the possible operating modes of the considered system in relation to the values taken by the inputs and avoiding excessive number of combinations (and thus *inference rules*) to be defined. As there are two input variables, each of which can have three different membership functions, there are 3^2=9 combinations, i.e. nine rules. Thus, the output variable was assigned nine different membership functions, one for each rule. The membership functions here chosen for the input variables were triangular and trapezoidal, as they are the most suitable for representing the type of inputs being examined. The structure of the inference rules, indicating the control action to apply according to all the possible combinations of the input variables, is:

If Collisions IS ... AND Throughput IS ... THEN VarRP IS ...

For the output variable, here *singleton* membership functions were chosen, i.e. each set comprises a single point with a degree of membership of 1 (*crisp*). It has to be noticed that the fuzzy controller output is not a single value among the nine defined in the rule set, as the fuzzy controller interpolates, according to fuzzy arithmetic, all the nine crisp output variables. The fuzzy controller was tuned using heuristic *trial and error* procedures.

3. Experimental evaluation

The behaviour of the *fuzzy* smoothing was investigated in a real scenario comprising 11 workstations equipped with the Linux O.S. and connected via a 10BASE-T Ethernet. The collision domain diameter is 10-metre. We implemented both the fuzzy smoother and the HIMD one, and performed a performance comparison in the same environment and operating conditions outlined in [3]. In both cases, the

smoothing driver was activated on each node with an RP_{max} of 100 ms, an RP_{min} of 3 ms and a τ of 1 ms; the observation period for the *sniffer* process used to measure the network load was set to 10 ms. As in [3], here we measured the *roundtrip delay* for the RT control messages exchanged between two processes, called Client and Server, running on two different PC_S, and calculated the deadline miss ratio for RT messages, taking the *deadline* for a complete transaction to be 129,6 ms. The duration of RT transactions was measured with a growing workload, progressively activating the processes generating NRT bursts, called the Station processes. In [7] a complete description of the experimental scenario is given. Here, for the sake of brevity, we just show the results for comparative purposes. Figs. 3 and 4 show how the roundtrip delay and the throughput are distributed with a varying *workload* during bursts using HIMD smoothing. The five bursts are of increasing intensity, ranging from only one NRT Station active to five NRT Stations active. Whereas roundtrip delay values are quite low in the absence of bursts, they increase considerably during bursts and several messages miss their deadlines. It can be also observed that the delays affecting RT messages are high even between one burst and another. This is confirmed by the throughput curve (Fig. 4), which is lower but broader than the workload one. This is because the HIMD approach delays the handling of NRT traffic beyond the end of the burst, thus keeping the throughput high for a certain time after the burst.

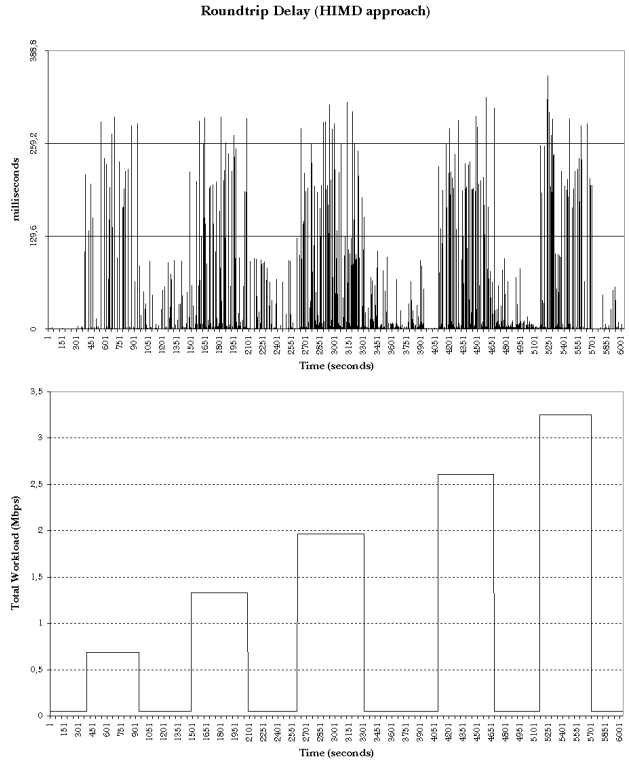


Fig. 3. Roundtrip delay for RT messages (HIMD)

Fig.3 shows that several RT messages feature high roundtrip delay values even after the NRT workload burst has ended, because a number of NRT messages have not been dispatched, so the overall traffic is still high.

The *fuzzy* controller used in this experiment comprised the *inference rules* shown in Table 1. Fig. 5 gives the roundtrip values for the single messages with varying workloads from all the processes, both RT and NRT, obtained using the fuzzy smoother. Comparison with Fig. 3 shows that here the delays are much shorter, even when there are bursts. Very few messages feature roundtrip delay values over 129.6 ms. This means that only a few messages collide with NRT traffic. As all the NRT traffic is dispatched during bursts, RT messages outside burst periods are not affected by an appreciable delay.

The improvement in performance achieved by fuzzy smoothing as compared with the HIMD approach is also confirmed by the total network throughput graph in Fig. 6, which shows a more regular trend of total throughput vs. the workload and higher throughput values than the graph in Fig. 4. This is due to the fact that the fuzzy smoother does not totally block NRT traffic, which may reach values very close to the workload. Moreover, in the time interval between consecutive bursts there is no "tail" due to the previous burst, because all the traffic has been handled on time. Thus, unlike the HIMD case, the effect of the bursts does not spread out the burst period. Finally, we compared deadline miss ratio (Dmr) obtained with the two approaches. With a *deadline* of 129,6 ms, the Dmr using the fuzzy smoother never exceeds 0,7%, about 1/3 of the one obtained using HIMD. This result is highly satisfactory for many soft RT applications.

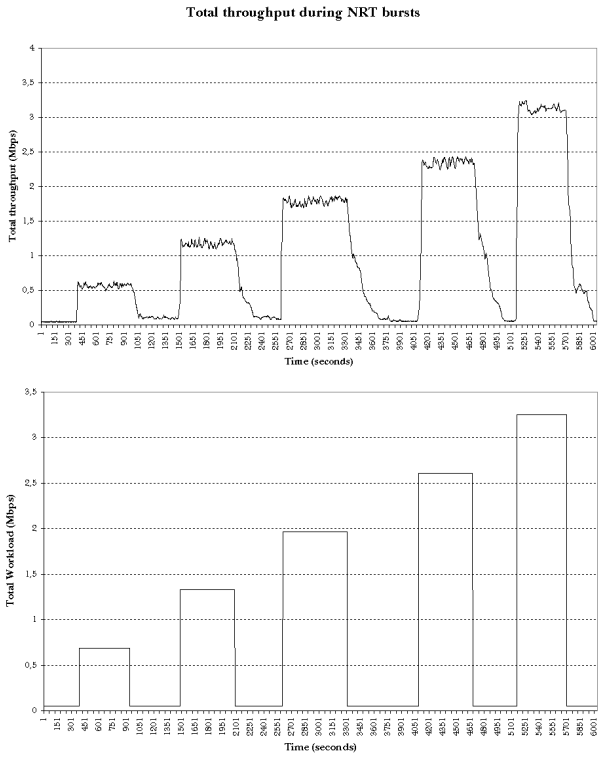


Fig.4. Total network throughput (HIMD)

Table. 1. The inference rules used

ANTECENDENT	CONSEQUENT
If Collisions IS Low AND Throughput IS Low THEN *VarRP* IS	$-RP_{min}$ *0.5
If Collisions IS Low AND Throughput IS Med THEN *VarRP* IS	$-RP_{min}$*0.3
If Collisions IS Low AND Throughput IS High THEN *VarRP* IS	0
If Collisions IS Med AND Throughput IS Low THEN *VarRP* IS	$-RP_{min}$*0.1
If Collisions IS Med AND Throughput IS Med THEN *VarRP* IS	0
If Collisions IS Med AND Throughput IS High THEN *VarRP* IS	$+RP_{max}$*0.2
If Collisions IS High AND Throughput IS Low THEN *VarRP* IS	$+RP_{max}$
If Collisions IS High AND Throughput IS Med THEN *VarRP* IS	$+RP_{max}$*0.7
If Collisions IS High AND Throughput IS High THEN *VarRP* IS	$+RP_{max}$*0.6

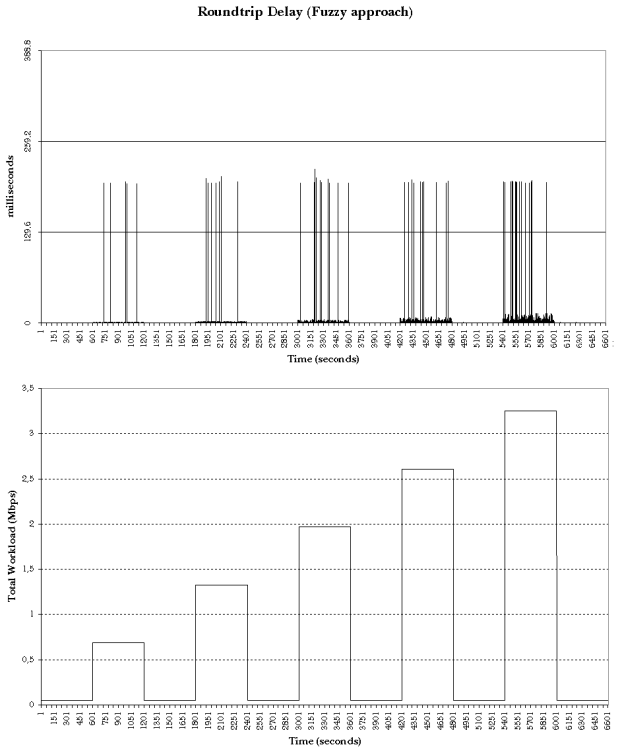


Fig. 5. Roundtrip delay for RT messages (fuzzy)

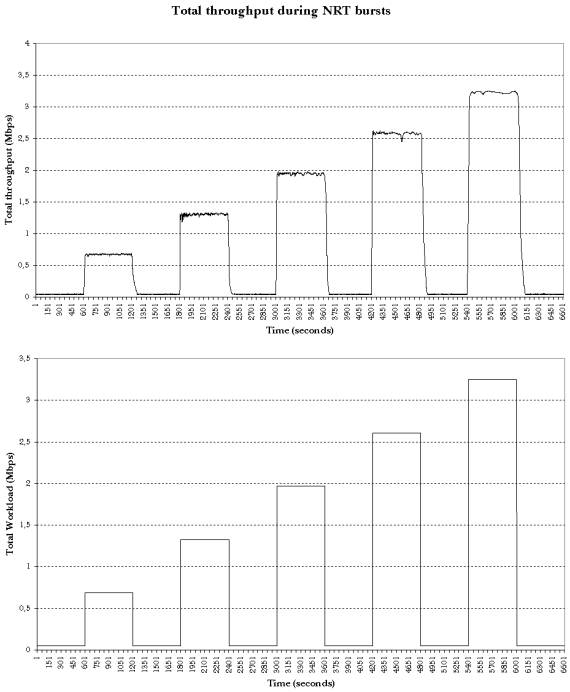


Fig. 6. Total network throughput (fuzzy)

4. Conclusions

Experimental results obtained in a real environment have confirmed the benefits of the approach proposed.

References

[1]S. Kweon, K.G. Shin, Q. Zheng, "Statistical Real-Time Communication over Ethernet for Manufacturing Automation Systems", *Proc. of RTAS'99*, June 1999, Vancouver, Canada.

[2]L. Lo Bello, M. Lorefice, O. Mirabella, S. Oliveri, "Performance Analysis of Ethernet Networks in the Process Control", *Proc. of IECON'00,* Puebla, Mexico, Dec. 2000.

[3] S. Kweon, K. G. Shin, et al. "Achieving Real-Time Communication over Ethernet with Adaptive Traffic Smoothing", in *Proc. of RTAS 2000*, pp.90-100, Washington DC, USA, June 2000.

[4]Yager R., Zadeh L.A. Editors, *An Introduction to Fuzzy Logic Applications in Intelligent systems*, Kluwer Academic, 1992.

[6] R.L. Cruz, "A Calculus for network delay, Part I: Network Elements in Isolation", *IEEE Trans. on Information Theory*, 37(1), Jan 1991.

[7] A. Carpenzano, R. Caponetto, L. Lo Bello, O. Mirabella, "Fuzzy Traffic Smoothing: an Approach for Real-time Communication over Ethernet Networks", to appear on *Proc. of WFCS'02,* Västerås, Sweden, August 2002.

A Multipoint Communication Protocol based on Ethernet for Analyzable Distributed Real-Time Applications

José María Martínez, Michael González Harbour, and J. Javier Gutiérrez

Departamento de Electrónica y Computadores
Universidad de Cantabria
39005-Santander, SPAIN
chema@gmx.net, mgh@unican.es, gutierjj@unican.es

Abstract

This paper presents a work-in-progress design and implementation of a software-based token-passing Ethernet protocol for multipoint communications in real-time applications, that does not require any modification to existing Ethernet hardware. Because the protocol is based on fixed priorities, applications using it can be easily modeled using common techniques for fixed priority systems, and well-known schedulability analysis techniques can be applied. We call this protocol RT-EP (Real-Time Ethernet Protocol).

1. Introduction[1]

Ethernet is by far the most widely used local area networking (LAN) technology in the world today. Unfortunately, Ethernet uses a non-deterministic arbitration mechanism (CSMA/CD) which makes it unsuitable for real-time communications. Several approaches and techniques have been used to make Ethernet deterministic in order to take advantage of its low cost and higher speeds than those of real-time field buses available today (like the CAN bus [9], for example). Some of these approaches are the modification of the Medium Access Control [6], the addition of transmission control [5], a protocol using time-triggered traffic [3], or the usage of a switched Ethernet [7].

Our research group has been working in the last few years on the development of MaRTE OS [1], a real-time kernel for embedded applications. The objective of this work is to add a real-time communication network to MaRTE. We want to achieve a relatively high speed mechanism for real-time communications at a low cost, while keeping the predictable timing behavior required in distributed hard real-time applications. The communications protocol proposed in this work can be classified as an addition of a transmission control layer over Ethernet, since it is basically a token-passing protocol in a bus [8].

1. This work has been funded by the *Comisión Interministerial de Ciencia y Tecnología* of the Spanish Government under grant TIC99-1043-C03-03

The paper is organized as follows: Section 2 describes how the communication protocol works. Section 3 gives details about the implementation and the model describing the timing behavior of its implementation. In Section 4 we provide some results on the overhead introduced by this protocol. Finally, Section 5 gives our conclusions.

2. Description of the Communication Protocol

RT-EP has been design to avoid collisions in the Ethernet media by the use of a token. Each station (processing node or CPU) has a transmission queue, which is a priority queue where all the packets to be transmitted are stored in priority order. Each station also has a set of reception queues that are also priority queues. Packets with the same priority are stored in FIFO order. The number of reception queues can be configured depending on the number of application threads (or tasks) running in the system and requiring reception of messages. Each application thread should have its own reception queue attached. The application has to assign a number, the channel ID, to each application thread that requires communication through the protocol. This channel ID is used for the purpose of identifying communication endpoints in a given station.

The network is logically organized as a ring. Each station knows which other station is its predecessor and its successor, so the logical ring can be built. The protocol works by rotating a token in this logical ring. The token holds information about the station having the highest priority packet to be transmitted and its priority value. The network operates in two phases. The first phase corresponds to the priority arbitration, and the second phase to the transmission of an application message.

For the transmission of one message, an arbitrary station is designated as the *token_master*. During the priority-arbitration phase the token travels through the whole ring, visiting all the nodes. Each station checks the information in the token to determine if one of its own packets has a priority higher than the priority carried by the token. In that case, it

changes the highest priority station and associated priority in the token information; otherwise the token is left unchanged. Then, the token is sent to the successor station. This process is followed until the token arrives at the *token_master* station, finishing the arbitration phase.

In the message-transmission phase the *token_master* station sends a message to the station with the highest priority message, which then sends the message. The receiving station becomes the new *token_master* station.

The maximum information size held by a packet is limited by the information field in an Ethernet frame, and it can vary from 0 to 1492 bytes) [2]. Fragmentation of messages at this layer is not allowed.

3. Implementation and Modeling of RT-EP

The functionality and architecture of the multipoint communication protocol are shown in Figure 1. This protocol offers three functions to any application using the network: *send_info* (to send a message), *recv_info* (to receive a message), and *init_comm* (to initialize the network). The application threads encapsulate the information in a message type, which is used both for transmission and reception. This message type contains the destination station address, the destination channel ID, and the priority of the message. When a message is sent, it is stored in the priority queue on the transmitting station. There is only one thread, the *Main Communication Thread*, that is responsible of reading the packets from the transmission queue and of writing the received packets into the reception queues. The highest priority packet to be transmitted determines the priority used for the priority arbitration token, as we described in the previous section.

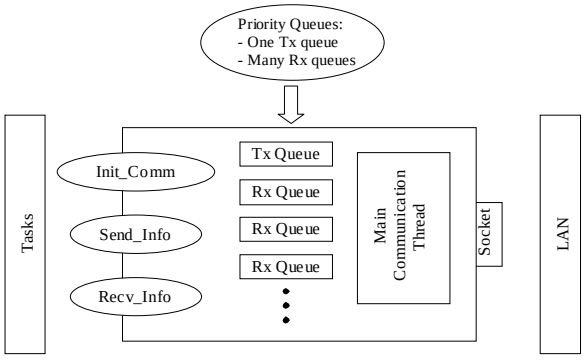


Figure 1. Functionality and details of RT-EP

The protocol has been implemented in GNU/Linux, directly over the network link layer, to test the design and to have a quick estimation of the overheads. In the near future the protocol will be implemented in MaRTE OS. We use RT-EP-token-packets to send the tokens and RT-EP-info-packets to send the information. To identify the stations, RT-EP uses the ethernet MAC addresses. There is no need for routing the information since we are working in a local area network.

In order for this protocol to work, the maximum number of communicating threads running in the system must be known at configuration time. This is the usual case in this kind of real-time system.

3.1. RT-EP Frame Formats

The frame formats used in our protocol goes into an Ethernet frame, which for Ethernet II has the following structure [2]:

8 bytes	6 bytes	6 bytes	2 bytes	46-1500 bytes	4 bytes
Preamble	*Destination Address*	*Source Address*	*Type*	*Data*	*Frame Check Sequence*

The *Type* field identifies what type of high-level network protocol is being carried in the data field. We use a value of 0x1000 for the *Type* field, which represents an unused number protocol, which could be changed if the protocol is registered in the future.

RT-EP packets are carried into the *Data* field of the Ethernet frame, that must be at least 46 bytes long. Due to this restriction, even though our packets can be less than 46 bytes long, a 46 bytes data field will be built. Our protocol has two types of packets:

- *Token Packet*: it is used to transmit the token and has the following structure:

1 byte	1 byte	2 bytes	6 bytes	34 bytes
Packet Identifier	*Token Identifier*	*Priority*	*Station Address*	*Extra*

  The *Packet Identifier* field is present also in the *Info Packet* and is used to identify the type of the packet. It can hold two different values for this type of packet: *Token* (used in the arbitration phase to get the highest priority packet) or *Transmit Token* (it grants the destination station permission to transmit a message). The *Token Identifier* will be used in the future to handle the loss of tokens. The *Priority* indicates the highest priority element on the LAN at the rotation time. The *Station Address* stores the address of the station with the highest priority packet. Finally, the 34 *Extra* bytes are needed to be compliant with the Ethernet protocol.

- *Info Packet*: it is used to transmit data and has the following structure:

1 byte	1 byte	2 bytes	2 bytes	2 bytes	0-1492 bytes
Packet Identifier	*Reserved*	*Priority*	*Channel ID*	*Info Length*	*Info*

The *Packet Identifier* has a value corresponding to an *Info Packet*. There is one *Reserved* byte for further use. The *Priority* field holds the priority of the packet being transmitted. The *Channel ID* is used to identify the destination queue in the destination station. The *Info Length* is the size of the data stored on the *Info* field. If the information to be transmitted is less than 38 bytes long, padding is performed in order to get the 46 data bytes required in an Ethernet frame.

3.2. RT_EP as a State Machine

RT-EP can be described as a state machine for each station in order to understand its functionality and obtain the relevant parameters for the different operations involved in the timing model. Figure 2 shows the states and the transitions between them, which are shortly described next:

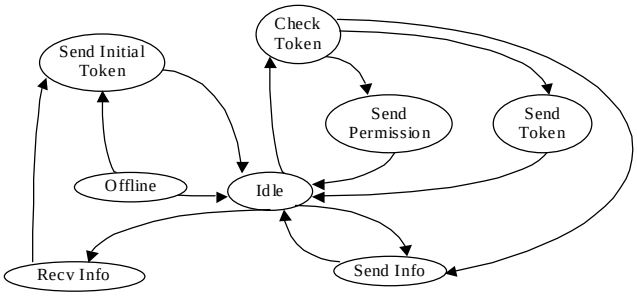


Figure 2. RT-EP state machine in each station

- *Offline*. It is the starting state reached during configuration time. Each station reads a configuration file describing the token ring and gets configured as one of its stations. The station configured as the initial *token_master* is set to the *Send_Initial_Token* state and the others are set to the *Idle* state.
- *Idle*. The station listens for the arrival of any packet. When a packet is received, a check is made to determine it's type: if it is an *Info Packet* the station switches to the *Recv_Info* state; if it is a *Token Packet*, two different states can be reached: *Send_Info* (if a *Transmit Token* is received), or *Check_Token* (when a regular *Token* is received).
- *Send_Initial_Token*. The station reaching this state becomes the *token_master*. A token is sent to the successor station, and the current station switches to the *Idle* state.
- *Check_Token*. If the station isn't the *token_master* the *Send_Token* state is reached; if it is, it switches to the *Send_Info* state if the *Station Address* is the current station, or to the *Send_Permission* state when not.
- *Send_Token*. The station compares the priority of the token with the highest priority element on its transmission queue, updates the token if its own priority is higher, and sends the token to next station. Then it switches to the *Idle* state.
- *Send_Permission*. The *token_master* role is lost and a the *Transmit Token* is built and sent to the highest priority station.
- *Send_Info*. This is the state in which a station has the highest priority packet on the ring and it is allowed to transmit it.
- *Recv_Info*. The information is written into the appropriate reception queue and the station switches to the *Send_Initial_Token* state, becoming the *token_master.*

3.3. MAST Model of RT-EP

This subsection draws out the modeling information of RT-EP according to MAST (Modeling and Analysis Suite for Real-Time Applications) [4]. This methodology provides an open source set of tools that enables engineers developing real-time applications to check the timing behavior of their application, including schedulability analysis for checking hard timing requirements. MAST includes the model of a fixed priority network as a specialized class of a processing resource. The model of the network encapsulates the relevant information to ensure that the schedulability analysis can be performed. In addition, MAST defines the network drivers, with parameters that represent the overheads of the activities executed by the processors to manage the communication packets.

We can use the MAST model to characterize RT-EP by obtaining the specific values for the network parameters. In order to have a complete description of the RT-EP model we must extend MAST by adding a new network driver (based on the existing *packet_driver*) which includes the operations to send and receive packets performed by the Main Communication Thread, the thread itself, and the protocol operations to manage and pass the tokens. The complete information to model RT-EP is described next using the MAST notation.

RT-EP Packet Driver. It is a specialization of a packet driver in which there is an additional overhead associated to passing the token. Its main attributes are:

- *Packet Send Operation*. It corresponds to the code executed in the *Idle* state followed by the *Send_Info* state.
- *Packet Receive Operation*. It corresponds to the code executed in the *Idle* state followed by the *Recv_Info* state and by the *Send_Initial_Token* state.
- *Number of Stations*.
- *Token Manage Operation*. Time required to send the token in the *Send_Token* or the *Send_Permission* states.

- *Token Check Operation*. Code executed in the *Idle* state followed by the *Check_Token* state.

The following attributes are used to characterize the *Fixed_Priority_Network* resource for RT-EP:

- *Max Packet Transmission Time* and *Min Packet Transmission Time*. They include only the time spent to send information bytes (1500 or 46 bytes).
- *Packet Overhead*. This is the overhead caused by the protocol information that needs to be sent before or after each packet. It is calculated as:

  (N+1)*(Min_PTT+EPB+TCO+TMO)

  which is the time spent to send a number of tokens equal to the number of stations, *N*, performing a complete circulation of the *Token*, plus one *Transmit Token*. The time to send a token is calculated as the sum of the *Min Packet Transmission Time*, Min_PTT, the time to send the Ethernet Protocol Bytes, EPB, the time of the *Token Check Operation*, and the time of the *Token Manage Operation*.
- *Max Blocking*. The maximum blocking caused by the non preemptability of message packets. In RT-EP, it is calculated as:

  N*(Min_PTT+EPB+TCO+TMO) + PWO+Max_PTT

  that represents a complete circulation of the token (*N* tokens sent), plus the blocking effect caused by the transmission of a lower priority packet (the *Packet Worst Overhead* and the *Maximum Packet Transmission Time*).

4. Overhead estimation

We have measured the CPU overheads of the protocol under GNU/Linux. Since this isn't a real-time OS, we have taken average values of our measurements. The times have been measured with a platform composed of two PCs (a Pentium 200 MMX and a Pentium 233 MMX) running GNU/Linux and connected by means of a null cable at 10 Mbps. The following table shows the average execution times of the operations involved in each state of the state machine description:

Operation	Time (µs)
Idle State	11.00
Send_Initial_Token	120.78
Check_Token	5.97
Send_Permission	107.61
Send_Token	99.42
Send_Info	149.70
Recv_Info	134.82

5. Conclusions

We have presented an implementation of a software-based token-passing Ethernet protocol for multipoint communications in real-time applications, that does not require any modification to existing Ethernet hardware. The protocol is based on fixed priorities and thus common tools for fixed priority schedulability analysis can be used. A precise timing model of the protocol has been obtained, which enables us to perform a schedulability analysis of a distributed application using this protocol.

Future work plans for this protocol are to extend it to take into account three kinds of failures: failure of a station, loss of a packet, or delay in handling a packet that could result in the duplication of a token. In addition, we have to port it to MaRTE OS.

References

[1] M. Aldea and M. González. “MaRTE OS: An Ada Kernel for Real-Time Embedded Applications”. Proceedings of the International Conference on Reliable Software Technologies, Ada-Europe-2001, Leuven, Belgium, Lecture Notes in Computer Science, LNCS 2043, May, 2001.

[2] Charles E. Spurgeon *Ethernet: The definitive Guide*. O’Reilly Associates, Inc. 2000.

[3] Paulo Pedreiras, Luis Almeida, Paolo Gar. “The FTT-Ethernet protocol: Merging flexibility, timeliness and efficiency”. Proceedings of the 14th Euromicro Conference on Real-Time Systems, Vienna, Austria, June 2002.

[4] M. González Harbour, J.J. Gutiérrez, J.C. Palencia and J.M. Drake: “MAST: Modeling and Analysis Suite for Real-Time Applications”. Proceedings of the Euromicro Conference on Real-Time Systems, Delft, The Netherlands, June 2001

[5] Chiueh Tzi-Cker and C. Venkatramani. “Fault handling mechanisms in the RETHER protocol”. Symposium on Fault-Tolerant Systems, Pacific Rim International, pp. 153-159, 1997.

[6] Jae-Young Lee, Hong-ju Moon, Sang Yong Moon, Wook Hyun Kwon, Sung Woo Lee, and Ik Soo Park. “Token-Passing bus access method on the IEEE 802.3 physical layer for distributed control networks”. Distributed Computer Control Systems 1998 (DCCS'98), Proceedings volume from the 15th IFAC Workshop. Elsevier Science, Kidlington, UK, pp. 31-36, 1999.

[7] Choi Baek-Young, Song Sejun, N. Birch, and Huang Jim. “Probabilistic approach to switched Ethernet for real-time control applications”. Proceedings of Seventh International Conference on Real-Time Computing Systems and Applications, pp. 384-388, 2000.

[8] ANSI/IEEE Std 802.4-1990. “IEEE Standard for Information technology--Telecommunications and information exchange between systems--Local and metropolitan area networks--Common specifications--Part 4: Token-Passing Bus Access Method and Physical Layer Specifications”.

[9] K. Tindell, A. Burns, and A.J. Wellings, “Calculating Controller Area Network (CAN) Message Response Times”. Proceedings of the 1994 IFAC Workshop on Distributed Computer Control Systems (DCCS), Toledo, Spain, 1994.

Flexibility, Timeliness and Efficiency over Ethernet

Paulo Pedreiras[1], Luís Almeida
DET / IEETA – Universidade de Aveiro
3810-193 Aveiro, Portugal
pedreiras@alunos.det.ua.pt, lda@det.ua.pt

Abstract

This paper summarises the materials presented in [11] concerning the quest for real-time behaviour over Ethernet and the new protocol FTT-Ethernet. This paper then includes a discussion on the use of this new protocol on networks based on switches, as this is becoming the main architectural choice in LANs.

1 Introduction

In the Internet Age, many new services are being created everyday, some of which will require real-time support, e.g. voice over IP, videoconference, remote monitoring of control processes, streaming services. To fulfil the requirements of such services, proper Quality-of-Service control mechanisms must be used either by transit networks as well as by access networks. In the first case, there are a few technologies that already support such mechanisms, e.g. ATM and Frame Relay. In what concerns the latter type, the main technology used is Ethernet for which the development of efficient QoS control mechanisms is particularly relevant. This paper summarises the materials presented in [11] concerning the QoS aspects of timeliness (real-time behaviour), flexibility and efficiency, as well as a new protocol, FTT-Ethernet. Finally the paper discusses the advantages that may arise from using switches to support the new protocol.

2 Real-time and Ethernet

The quest for real-time behaviour on Ethernet led to several approaches and techniques. This section presents and characterizes some of these paradigmatic efforts that, nevertheless, either require specialised hardware, are suited to soft-real-time operation only, or are bandwidth or response-time inefficient.

2.1 Modification of the Medium Access Control

This approach consists on modifying the Ethernet MAC layer to achieve a bounded access time to the bus (e.g. [2], [7] and [8]). For instance, the solution presented in [2] (CSMA/DCR) consists in a binary tree search of colliding nodes, that is, there is a hierarchy of priorities. Whenever a collision happens the lower priority nodes voluntarily cease contending for the bus, and higher priority nodes try again. This process is repeated until a successful transmission occurs. Aspects against: requires modification of the firmware (no benefit from economy of scale of standard Ethernet), long worst-case transmission time with respect to average (pessimistic analysis thus bandwidth under-utilization).

2.2 Adding transmission control over Ethernet

This method consists on adding a layer above Ethernet, intended to control the instants of message transmissions, ending up with a bounded number of collisions or even a complete avoidance of them. In favour: standard Ethernet hardware can be used. Several different transmission control implementations are referred below.

Master/Slave. Any node transmits messages only upon receiving an explicit command message issued by one particular node called Master. In favour: relatively precise timeliness (depending on the master). Against: introduces a considerable protocol overhead (master messages); inneficient handling of event-triggered traffic (unknown transmission instants).

Token-Passing. A token is circulated among the nodes. Only that one currently holding the token is allowed to transmit and the token holding time is upper bounded (e.g. IEEE 802.4). Against: protocol overhead (bandwidth used by token); poor support for periodic traffic; bus inaccessibility caused by token losses.

Virtual Timed-Token. Basis of the RETHER protocol [3]. In real-time mode, nodes are divided in two groups: the RT group for nodes with bandwidth reservations; the NRT group for all the others. The real-time messages are assumed to be periodic, and time is divided in cycles with the duration of one time unit. Access to the channel for both kinds of traffic is regulated by a token. First, the token visits all the RT senders having messages to be produced in that cycle, and after the NRT nodes, if enough time is left until the end of the cycle. In

[1] This work was partially supported by the Portuguese Government through grant PRAXIS XXI/BD/21679/99 and project CIDER-POSI/1999/CHS/33139.

[1] This work was partially supported by the Portuguese Government through grant PRAXIS XXI/BD/21679/99 and project CIDER-POSI/1999/CHS/33139.

favour: online admission control of RT messages. Against: lack of support for real-time sporadic traffic; high overhead (as in master/slave).

TDMA. In this case, nodes transmit messages at pre-determined disjoint instants in time in a cyclic fashion. In favour: high bandwidth efficiency (no control messages). Against: requires precise clock synchronization; hardly supports dynamic changes in the message set (communication requirements are distributed and changes must be done globally).

Virtual Time Protocol. This protocol [9] [10] tries to reduce the number of collisions on the bus while offering the flexibility to implement different scheduling policies, e.g. minimum-laxity first. When a node wishes to transmit a message it waits for a given amount of time counting from the moment the bus became idle. This amount of time is calculated according to the desired scheduling policy. When that time expires, and if the bus is still idle, the node tries to transmit the message. If a collision occurs, then there is another node with a message with the same laxity. In this case the protocol uses a probabilistic approach: the nodes involved in the collision either retransmit the message with probability *p* or wait for another similar amount of time. Against: high sensitiveness to the proportional constant value used to relate the waiting time with the scheduling policy; with collisions, worst-case transmission time is much higher than average.

Window protocols. These types of protocols have been proposed for both CSMA/CD and token ring networks [9]. In the former ones, the nodes on a network agree on common time interval named window. The bus state is used to assess the number of nodes with messages to be transmitted within the time window: if the bus remains idle, there are no messages to be transmitted in the window; if only one message is in the window, it will be transmitted; if two or more messages are within the window, a collision occurs. Depending on the bus state, several actions can be performed: if the bus remains idle, the time window is increased; in the case of a collision, the time window is shortened. Against: collisions lead to bus under-utilization if timeliness must be guaranteed.

2.3 Traffic shaping

This technique follows an approach based on the statistical relationship between bus utilization and collision probability. Keeping the bus utilization below a given threshold allows obtaining a deemed collision probability. One implementation of this technique is presented in [6]. An interface layer, called traffic smoother, is placed between the transport layer (TCP/UDP) and Ethernet. The traffic smoother gives real-time traffic priority over non-real-time one (NRT) within each node. Moreover, this layer keeps track of the traffic generated by the node, and controls the transmission of NRT traffic, in order to keep the network load originated by the node below the specified value. This approach provides statistical guarantees, thus it is suited to support soft real-time traffic, only.

2.4 Switched Ethernet

The use of switches became very popular recently, as a way to improve the performance of shared Ethernet. Switches provide a private collision domain for each of its ports. When a node transmits a message, this one is received by the switch and then buffered in to the output ports where the receiver(s) of the message are connected. If several messages addressed to a given port arrive in a short interval, they are queued and then sequentially transmitted. Concerning the scheduling of messages waiting in an output port, 8 priority queues are available (IEEE 802.1D). The scheduling policy used at this level is a topic currently addressed in the scientific community (e.g. [5]).

Simply adding a switch to an Ethernet network is not enough to enforce real-time behaviour. For example, in distributed control systems the producer/ consumers model is typically used, in which one producer of a given datum (e.g. a sensor reading) sends it to several consumers of that datum. In shared Ethernet this feature is supported by means of special addresses (multicasts). However, switches handle this kind of traffic as broadcasts, and thus one of the major benefits of switched Ethernet, multiple simultaneous transmission paths, can be seriously compromised. Other problems concerning the use of switched Ethernet are [1]:

- In the absence of collisions the switch introduces an additional latency;
- The number of available priority levels is too small to support efficient priority based scheduling;
- The switch only makes Ethernet deterministic under controlled loads, therefore, to support hard real-time traffic an appropriate admission control policy must be added.

3 FTT-Ethernet protocol

The rationale behind the FTT-Ethernet protocol is to support hard real-time communication in a flexible and bandwidth efficient way, using COTS hardware. It aims at distributed real-time systems used either in industrial automation, multimedia or embedded control applications. The protocol is based on the Flexible Time-Triggered (FTT) paradigm, which can be implemented on different networks, e.g. FTT-CAN [4] that uses the Controller Area Network. Independently of the underlying network, FTT-based protocols present the following features:

1. Time-triggered communication with operational flexibility;
2. Support for on-the-fly changes both on the message set and the scheduling policy used;
3. Online admission control to guarantee timeliness to the real-time traffic;
4. Indication of temporal accuracy of real-time messages;
5. Support of different types of traffic: event-triggered, time-triggered, hard real-time, soft real-time and non-real-time;
6. Temporal isolation: the distinct types of traffic must not disturb each other;
7. Efficient use of network bandwidth;

Based on these features, table 1 shows a summarized comparison among several approaches to real-time communication over Ethernet, including FTT-Ethernet.

3.1 Protocol description

To support the features referred to above, the FTT-Ethernet protocol relies on centralized scheduling and master/ multi-slave transmission control.

Protocol	Traffic classes			Dynamic comm. req.	Timeliness guarant.	Temporal isolation	Efficiency	COTS hardware
	Real-time		Non-Real-time					
	Time Trig.	Event Trig.						
CSMD/DCR	No	Yes	Yes	Yes	Hard (1)	No	Low (2)	No (5)
TDMA	Yes	No	No	No	Hard	N.A.	High	Yes
Virtual Time Protocol	No	Yes	Yes	Yes	Hard (1)	No	Low (2)	Yes
Windows Protocols	No	Yes	Yes	Yes	Hard (1)	No	Low (2)	Yes
Virtual Timed-Token	Yes	No	Yes	Yes	Hard	Yes	Low (3)	Yes
Switched Ethernet	No	Yes	Yes	Yes	No (4)	No	High	Yes
Traffic Smoothing	No	Yes	Yes	Yes	Soft	No	Low (2)	Yes
FTT-Ethernet	Yes	Yes	Yes	Yes	Hard	Yes	High	Yes

(1) Worst-case resposnse time much higher than the average value
(2) Collisions are part of the protocol
(3) Each Real-time message is preceded by a control message
(4) Can be achieved by the use of a system admition control policy
(5) Requires modifications on the NIC's firmware.
N.A. Not appliable

Table 1: Comparing different approaches to real-time communication over Ethernet.

Centralized traffic scheduling allows having both the communication requirements and the message scheduling policy localized in one node, the Master, facilitating on-line changes to both. On the other hand, such centralization also facilitates the implementation of on-line admission control in the Master node.

Master/multi-slave transmission control allows enforcing the traffic timeliness in the bus without incurring in a high penalty concerning the efficiency in bandwidth utilization. The first aspect is typical of master-slave transmission control since the master explicitly tells each slave when to transmit, thus enforcing the traffic timeliness. The second aspect results from a single Master message triggering the transmission of several messages by slave nodes.

In FTT-Ethernet traffic is allocated in fixed duration time slots called Elementary Cycles (EC). The bus time is organized in an infinite succession of ECs. Each EC starts with a trigger message (TM), sent by the Master, and is composed by two sequential phases for transmission of time and event-triggered traffic (**Figure 1**). The traffic on the synchronous window is controlled by the TM, which contains the identification and length (Tx_i in **Figure 1**) of the messages that must be transmitted within the respective EC (EC-schedule).

Each node holds a table identifying which synchronous messages it produces. Upon reception of the EC trigger message, slave nodes decode the EC-schedule information and compare it with the table of the locally produced messages, in order to identify which synchronous messages the node should produce in the current EC. These messages are then queued for transmission in the synchronous window. The information carried by the TM is enough to allow producer nodes to transmit the messages at disjoint time instants, therefore collisions are avoided. The transmission of synchronous messages is handled by the synchronous messaging system (SMS).

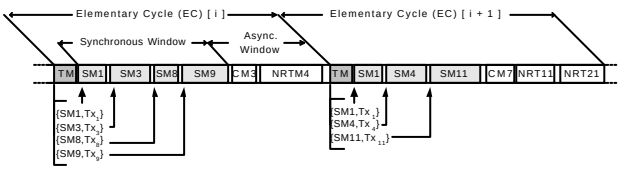


Figure 1. Elementary Cycle structure

The FTT-Ethernet protocol supports asynchronous traffic for event-triggered communication. The FTT-Ethernet protocol receives and queues the transmission requests originated in the application layer. Nodes are periodically pooled for asynchronous messages, and, when allowed to, the respective messages are de-queued and transmitted.

The FTT-Ethernet also supports non-real-time traffic, which is transmitted within the asynchronous window. This type of traffic is handled under a best effort policy and is typically associated to common applications using higher-level communication protocols such as TCP/IP (e.g. http, ftp). Real-time asynchronous traffic is transmitted first and then, if time is available within the EC, nodes producing non-real-time messages are polled. The polling order can be scheduled in order to support distinct Quality of Service (QOS) according to nodes requirements. Asynchronous message transmission is handled by the Asynchronous Messaging System (AMS).

3.2 The Master node

The Master node plays the role of system coordinator and it is responsible for keeping a database holding the system configuration and communication requirements (SRDB); building EC-schedules according to the particular scheduling policy implemented; and finally broadcasting the EC trigger message, containing these schedules, at the start of each EC.

A set of tasks residing in the Master node implement the functionalities required for proper system operation, namely system configuration and management, message scheduling and dispatching and admission control.

3.3 Slave nodes

Slave nodes execute the application software required by the user, eventually requesting the services delivered by the communication system. This system is organized as two parallel stacks, one for non-real-time and the other for real-time communication. The former uses a standard IP protocol suite, where the only specific component of the FTT-Ethernet protocol is a modified Data Link Layer (DLL). The latter follows the collapsed 3 layers OSI reference model typically found in fieldbus systems, providing a specific application interface (Real-Time Application Programming Interface), which enables the applications to configure the messages locally produced and consumed, respectively update or read the value of such real-time entities and set-up call-backs associated to communication events.

In what concerns the DLL, a transmission control layer (FTT-Ethernet Interface Layer) is added on top of the Ethernet layer, both for real-time and non-real-time communication. The FTT-Ethernet Interface Layer receives and decodes the TM and transmits the locally produced messages according to the respective EC-schedule. Moreover,

this layer also handles received data messages, passing the data to the respective protocol stack when the data is locally consumed.

4 Shared or Switched Ethernet

As stated in section 3.1, the FTT-Ethernet protocol performs collision-free message exchange over shared Ethernet, since all messages are scheduled at disjoint time instants. In this scenario, each node must decode the temporal information conveyed in the trigger message. This information is then used to set up timers that will trigger the message transmissions at appropriate time instants.

Despite having been originally designed for shared Ethernet, the new protocol can also work without any modification over switched Ethernet or over mixed shared/switched networks. However, in the absence of shared segments, i.e. relying on switches only, the FTT-Ethernet protocol can take advantage of switched Ethernet features. For example, the queueing at the switch ports can be used to alleviate the processing overhead required in each node because there is no need for tight control on the message transmission instants. In this case, upon reception of the trigger message, the nodes only need to identify which messages they should produce within the EC, and transmit them immediately. The switch prevents destructive collisions and serializes the message transmissions in the output port queues. The fact that this option requires networks built exclusively with switches is not particularly restrictive since that is fast becoming the preferential architectural choice in current LANs.

On the other hand, the use of FTT-Ethernet may even contribute to reduce some of the problems of using switches in real-time applications as referred to in section 2.4. This comes from the fact that FTT-Ethernet performs the traffic control required to support adequate management in the output port queues and enforcing priority scheduling beyond the priority levels available in 802.1D.

In the former case, the Master can schedule transmissions taking into account the destination address of messages, either broadcast, multicast or unicast. This knowledge can be used in suitable scheduling policies for two different purposes. On one hand, it can be used to avoid scheduling more messages per EC to each output port than those that fit in the respective queue, thus preventing overflow and the consequent loss of messages. On the other hand, this knowledge can also be used to promote the simultaneous scheduling of messages that follow disjoint paths, taking advantage of the possibility of simultaneous data transmissions in the switches and thus improving the overall throughput.

In the latter case, the scheduling carried out by the Master node may take into account individual priorities of each message, possibly dynamic priorities, e.g. for EDF scheduling, which are neither restricted nor correlated to the 8 priority levels defined in 802.1D. This way, FTT-Ethernet supports strict priority scheduling within each of the priority levels defined in the standard. The term *strict*, though, can only be applied at a coarse time scale, in EC units, since priority invertions within the EC can occur.

Finally, using FTT-Ethernet on hierarchical multi-switch networks requires the addition of extra idle-time in each EC to cope with the forwarding delays at each level. Notice that all the traffic related to a given EC must arrive to the farthest point in the network within the same EC as perceived by the Master.

5 Conclusion

The FTT-Ethernet protocol presented in [11] is more flexible and efficient than other existing techniques to enforce real-time behaviour on Ethernet (table 1). Furthermore, its application to switched networks has been discussed and seems particularly advantageous. Therefore, the new protocol seems well adapted to enforce the QoS control mechanisms required by emergent applications, even Internet-based.

6 References

[1] Decotignie, J-D. A perspective on Ethernet as a Fieldbus. *FeT'01, 4th Int. Conf. on Fieldbus Systems and their Applications*. Nancy,France. Nov. 2001.

[2] LeLann, G, N. Rivierre. Real-Time Communications over Broadcast Networks: the CSMA-DCR and the DOD-CSMA-CD Protocols. *INRIA Report RR1863*. 1993.

[3] Venkatramani, C., T. Chiueh. Supporting Real-Time Traffic on Ethernet. *IEEE Real-Time Systems Symposium*. San Juan, Puerto Rico. Dec 94.

[4] Almeida, L., P. Pedreiras and J.A. Fonseca. FTT-CAN: Why and How. *IEEE Trans. on Industrial Electronics* (to appear). December 2002.

[5] Jasperneit, J., P. Neumann. Switched Ethernet for Factory Communication. *ETFA'01, IEEE Conf. Emerging Techn. on Factory Automation*. Antibes, France. Oct. 2001.

[6] Kweon, S-K. and K. G. Shin. Achieving Real-Time Communication over Ethernet with Adaptive Traffic Smoothing. *IEEE Real-Time Technol. and Applications Symp*., 90-100. Washington DC, USA. June 2000.

[7] Shimokawa, Y. and Y. Shiobara. Real-Time Ethernet for Industrial Applications. *IECON'85*, pp829-834. 1985.

[8] Court, R.. Real-Time Ethernet. *Computer Communications*, **15** pp. 198-201. April 1992.

[9] Malcolm, N., W. Zhao. Hard Real-Time Communications in Multiple-Access Networks. *Real Time Systems* **9**, 75-107. Kluwer Academic Publishers. 1995.

[10] Molle, M., L. Kleinrock. Virtual Time CSMA: Why two clocks are better than one. *IEEE Transactions on Communications*. **33(9)**:919-933. 1985.

[11] Pedreiras, P. L. Almeida and P. Gai. The FTT-Ethernet Protocol: Merging Flexibility, Timeliness and Efficiency. *ECRTS'02, EUROMICRO Conf. on Real-Time Systems*, Vienna, Austria. June 2002.

SBM protocol for providing real-time QoS in Ethernet LANs

Anis KOUBAA Aref JARRAYA Ye-Qiong SONG

LORIA – UHP Nancy 1 - INPL - INRIA Lorraine
2, av. de la Forêt de Haye
54516 Vandoeuvre – France

Email : akoubaa@loria.fr; ajarraya@ensem.inpl-nancy.fr; song@loria.fr

Abstract *- This paper deals with the performance evaluation of LAN-Integrated Service protocol called SBM (Subnetwork Bandwidth Manager), a solution to handle QoS requirements over Local Area Networks. SBM is an RSVP-based protocol, which consists in electing a manager over a LAN segment to map RSVP-flows into an appropriate class of service and handles admission control and bandwidth reservation operations for such flows. To show how SBM is useful for guaranteeing requested quality of service for real-time admitted flows, we simulated the bandwidth reservation and message scheduling in an Ethernet switch for different input flows sharing a same output trunk link. DSBM[1] election has also been simulated in order to evaluate time for DSBM failure recovery over switched and shared LAN topology.*

1. Introduction

With the emergence of bandwidth-greedy and/or time-sensitive applications, the need of guaranteed QoS (Quality of Service) for these applications becomes of prime importance in the underlying networks. For this purpose, many approaches have been developed so far to provide real-time QoS guarantees for time-sensitive applications. In the Internet community, the two widespread approaches are *IntServ* and *DiffServ***.** *IntServ* makes use of RSVP protocol with a bandwidth reservation in the routers and the related end-hosts along the path of IP packets to guarantee the end-to-end delay. For scalability reason, in *DiffServ*, an end user just needs to mark in the DS field of each of its packets the desired QoS class to signaling to routers its QoS demand (PHB: Per Hop Behaviour). Unlike *DiffServ*, which provides a per-class guarantee, *IntServ* provides a per-flow guarantee, which may arise the scalability problem in the Internet, but can be suitable to the industrial LAN context where the number of simultaneous flows to be handled should not be very important.

Recently, there is a willing to use Ethernet and its *de facto* upper layer protocols (TCP/IP and standards Internet applications) for factory communications. Although switched Ethernet can be configured to provide real-time QoS at the data link layer [4][5], for being able to take advantages of the upper layer Internet standards, protocols like *IntServ* or *DiffServ* must be deployed. The problem for deploying RSVP over Ethernet LANs is that RSVP stops at router level. To deal with this problem, an extension of IntServ-RSVP called SBM was defined [1] for LAN usage, which is a signaling protocol for RSVP-based admission control over IEEE 802-style networks. It supports the mapping of RSVP-enabled flows to Ethernet LANs providing the required QoS defined by RSVP [8] parameters.

SBM operates as follows:

- **DSBM Election Mechanism**: This procedure leads to designate a manager for a group of LAN-interconnected stations to handle the QoS requests on the managed segment. The elected member is called DSBM for Designated Subnet Bandwidth Manager. The principle is similar to the election of the Root Bridge in IEEE spanning tree protocol. For fault tolerance, the failure of the current DSBM leads to re-election of another DSBM
- **Bandwidth Reservation**: In that case, a single DSBM will manage the resources for those segments treating the collection of such segments as a single managed segment for the purpose of admission control. A station that wishes to send a guaranteed flow over the managed segment must firstly send a request to DSBM, which decides if a bandwidth reservation could be achieved.

SBM is defined to be used in both shared and switched LANs. Nowadays, switched LANs are more and more popular, for this reason we have chosen, within this work, to simulate the performance of SBM protocol over switched Ethernet LANs.

The contribution of this paper is to give a design and simulation framework for performance evaluation of LANs running SBM protocol. The model was developed using OPNET [11] Software. We built a switched network running SBM protocol and evaluated the

[1] DSBM : *Designated Subnetwork Bandwidth Manager* which performs Bandwidth reservation for incoming flows

performance of SBM in terms of message response-time and DSBM re-election time for failure recovery. For showing the importance of bandwidth reservation to provide a fair service and guaranteed QoS for time-sensitive applications, a comparative study between static scheduling (FIFO, SPQ *(Strict Priority Queueing)*) and per-flow scheduling (WFQ [10], PGPS [7]) is done.

2. QoS over IEEE802.3 LANs overview

In this section, we present some QoS features deployed over Local Area Networks. More details can be found in [1], [2] and [3].

2.1 QoS Legacy over Ethernet LANs

Initially, IEEE802.3 style networks do not provide any quality of service guarantees for any kind of traffic. All frames cross networks in best-effort fashion having or not real-time requirements. CSMA/CD protocol for shared half-duplex link does not provide deterministic medium access delay. This is not suitable for real-time sensitive applications and bandwidth-greedy flows.

Enhancements have been achieved by bridging solution, which reduces collision domain size by micro-segmenting the shared segment. Fully switched topologies can give deterministic access delay for the MAC layer as every node has its dedicated link but introduce additional latency upon frame reception and forwarding comparing to hub-repeaters.

The extended Ethernet format supporting *user_priority* tag, defined by *IEEE802.1p/Q* enables traffic classification for IEEE802 style networks. The 3-bit sized *user-priority* field enables differentiation between 8 traffic classes from 0 for lowest priority to 7 for highest priority flows. This field could be used by switches, according to IEEE802.1D standard.

2.2 Bandwidth reservation

Solutions given in previous paragraph by standard do not give any recommendation on how to deploy and handle traffic classes over LAN topology. However, there is much work built for QoS guarantee over IP networks (Internet) and mainly the Intserv and Diffser IETF working groups propositions [8][9]. Though, there is not known standards for bandwidth management over LAN until the SBM proposition given by ISSSL IETF working groups, which defined a framework for bandwidth reservation and QoS handling over IEEE802 networks [1][2][3].

The main idea of this proposal is to use the work carried out by IntServ-RSVP working group and defines the mapping of RSVP and Integrated services onto specific subnetwork technologies. This leads to designate an elected manager for a given segment to make bandwidth reservation for real-time applications. Segment may be *(a)* a shared Ethernet or Token ring bus resolving contention for media access using CSMA or token passing, *(b)* a half duplex link between two stations or switches or *(c)* one direction of a switch full-duplex link. Once the manager is elected by the DSBM election protocol for a given segment, it would obtain information on available resources such as bandwidth of the managed segment. All RSVP-Based reservation requests that transit would be processed by DSBM before forwarding it over the shared segment. The beauty of this protocol is that it supports RSVP protocol, and its implementation does not require many changes to RSVP request processing. A complete description for processing requests and implementation guidelines are detailed in [1].

3. Bandwidth reservation over Ethernet

Traditional Ethernet networks don't use any kind of bandwidth reservation. Traffic, that transit LAN domain, cannot make resource reservation. In best case, real-time traffic could have some better processing within switches using priority-based scheduling such as SPQ. Naturally, in that case, as switch cannot handle more than 8 parallel queues [IEEE802.1p], traffic will be scheduled in *aggregate*. For example, all video streams would be processed within a single queue. Problems exist if there are many streams to be handled within a same priority, especially when real-time constraints are hard. A solution that can be achieved using SBM protocol is to make bandwidth reservation and perform per-flow guarantee with WFQ scheduling algorithm. The manager processes the RSVP reservation request and accepts them whenever there are enough resources.

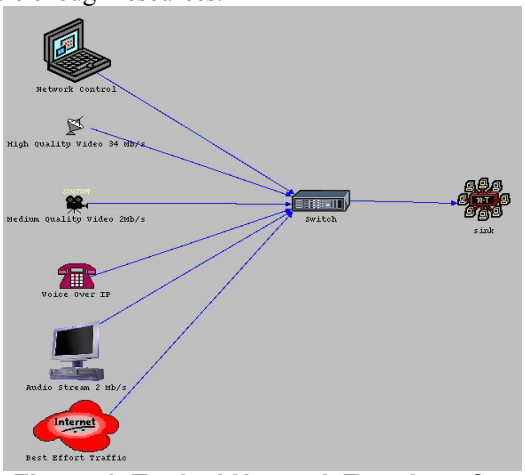


Figure 1. Typical Network Topology for Bandwidth Reservation

The following table shows flow characteristics for streams to be scheduled over the switch-manager.

Table 1. Flow Characteristics

Flow	Data Rate	Frame Length Distribution	Frame Inter-arrival Time (sec)
Network Control	20 kb	exp(1 Kb)	const(0,05)
HQ Video Stream	34 Mb	12 Kb	const(0,3336 * 10-3)
MQ Video Stream	2 Mb	6 Kb	const(2,86 * 10-3)
Voice	64 Kb	8 Kb	ON{exp(1), const(0,05)}/ OFF{exp(1,5)}
Audio	2 Mb	2 Kb	const(1,15 10^{-3})
Best Effort	1 Mb	Uniform(8 Kb, 12 Kb)	exp(0,01)

The total amount of traffic is about 41 Mb/s.

We have developed two scenarios to compare between these scheduling policies in terms of frame response time. The first one is highly loaded scenario with a total load of 0.9 and the second is an almost overloaded scenario with a load near to one (0.999). We further assume that the deadlines of the packets are equal to their periods (inter-arrival time). We mention that with SPQ scheduling, all video streams are handled within a same queue and same thing for audio streams. Here is the table for average and maximum response time for both scenarios.

Table 2. Flow Response Times with different scheduling algorithms

Flow	Deadline (ms)	Average Response Time (ms)						Maximum Response Time (ms)					
		Load 0.9			Load 0.999			Load 0.9			Load 0.999		
		FIFO	SPQ	WFQ	FIFO	SPQ	WFQ	FIFO	SPQ	WFQ	FIFO	SPQ	WFQ
Network Control	50	5,00	0,137	0,46	80	0,1635	9	170	1,357	11,251	387,0	1,611	348,40
HQ Video Stream	0,3336	5,00	0,31	0,294	80	0,4125	0,375	170	**2,569**	0,346	387,0	2,888	0,545
MQ Video Stream	2,86	5,00	0,31	0,325	80	0,4125	1,25	170	2,569	1,955	387,0	2,888	8,452
Voice	50	5,00	0,46	0,66	80	2,95	110	170	15,225	10,79	387,0	34,885	1213,7
Audio	1,15	5,00	0,46	0,221	80	2,95	0,41	170	**15,225**	1,112	387,0	34,885	2,982
Best Effort	10	5,00	175	0,6175	80	3800	17	170	**1899,1**	8,927	387,0	15344	222,40

It could be understood from results that WFQ gives better resource management. With resource reservation and per-flow scheduling, response time is better with WFQ than SPQ for Audio and Video streams. WFQ gives more fair scheduling behavior when the load is very high and can guarantee more narrow delays for lower priority flows without violation of higher priority ones. For example, from maximum response time results under a load of 0.9, WFQ meets the hard real time requirements of HQ and MQ video streams without violating the real time requirements of Network control traffic. Also In this case, all streams, even Best effort, meet their deadlines, which is not achieved with SPQ for the HQ Video Stream, Audio and Best Effort. This is explained by the advantage of per-flow scheduling and the ability to efficiently manage bandwidth resources.

An other fact, is when a congestion situation occurs, WFQ serve all flows with respect to their coefficient even that it does not provide all the requested bandwidth but does not make some flows to suffer from starvation as done by SPQ scheduler that serves only highest priority flows. This is explained by result for maximum response time with 0.999 of load.

Another advantage using SBM protocol is that it enables admission control so that a flow is admitted only if there are enough resources; else, it will be processed in best effort class. Moreover, a policy control could be used with SBM to prevent misbehaved sources from causing network congestion, which can affect the fairness of scheduling.

4. Modeling and performance evaluation of DSBM election algorithm

4.1 Motivation

Fault-tolerance is one of important issue for real-time application. In fact, when a manager is elected, it would manage resource reservation for real-time flows that need special processing to meet their real-time requirements. If the elected DSBM fails *(DSBMDeadIntervalTimer* fires), all the SBMs enter the

Elect state and start the election process. At the end of the election Interval, the elected DSBM sends out an *I_AM_DSBM* advertisement and the DSBM is then operational. This operation must not be too long to not disturb real-time behavior of current reserved flows. All reservations should be transferred to the new-elected DSBM. Next paragraph describes our SBM protocol model under OPNET simulator and results of time-to-recovery for switched and shared topology.

4.2 DSBM election Model

We implemented the DSBM election algorithm using OPNET simulation environment as described in [1].

The process model we developed will run on each Ethernet station, which communicates with each other with message. We define an SBM frame that will be used to store DSBM address and the priority of each SBM node. This information will be used by other stations to update their *LocalDSBMAddrInfo*, which represents in the end the information of the Best DSBM *(DSBM Address, DSBM Priority*). A complete description of model design and implementation is given in [6]. This process model treat two type of messages defined in [1]:

- ***DSBM_WILLING*** message sent by an SBM station to declare its candidacy to election process,
- ***I_AM_DSBM*** message sent by the DSBM itself for other SBM clients on the managed segment to declare itself as manager and to make sign of life every *RefreshIntervalTimer* period.

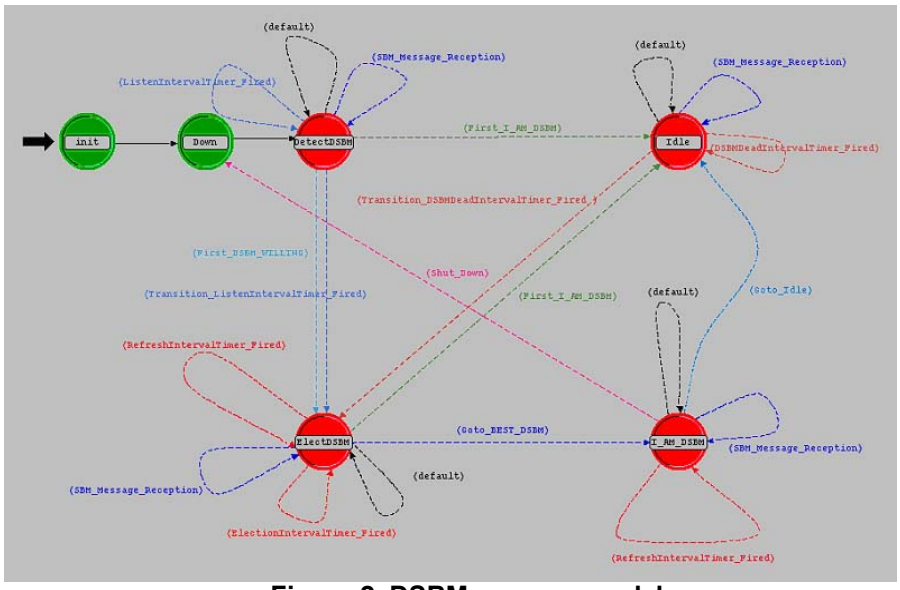


Figure 2. DSBM process model

4.3 Scenarios description and main results

We have run simulations for different Ethernet architectures, shared and switched.

In shared architecture, the elected DSBM would make resource reservation over managed segment, whereas in switched topology, DSBM could serve as a manager for the entire network when a centralized implementation of DSBM is used [3]. For both topologies, we have tried the election process with 2, 4, 8 and 16 SBM nodes to simulate the recovery-time, additional load resulting from sending *DSBM_WILLING* and *I_AM_DSBM* messages.

In all scenarios we have chosen these values for SBM Timers. There is recommendation for *ElectionIntervalTimer* in [1] to be set to 5 seconds. Suggestions are made for other timers but no recommended values.

Table 3. SBM Timers

Timer	Values
ListenIntervalTimer	3
DSBMDeadIntervalTimer	15
ElectionIntervalTimer	15
RefreshIntervalTimer	5

Simulations show the effect of increasing number of SBM candidates for DSBM election process. We collect through trace files, time needed for each station to make knowledge of the elected DSBM. In fact, this time is less that *ElectionIntervalTimer*. Once all stations know the elected DSBM, only the latter will continue to send *DSBM_WILLING* messages until *ElectionIntervalTimer* has to be fired.

The table below shows the time needed to discover the DSBM *i.e.* the instant from which only the DSBM sends *DSBM_WILLING* advertisements. Actually, all SBM stations know the DSBM, but declaration is done only after *ElectionIntervalTimer* expiration.

Table 4. Time-to-recovery (ms)

Topology	Shared	Switched
2 Nodes	0.059	0.024
4 Nodes	0.871	0.232
8 Nodes	3.981	1.612
16 Nodes	6.511	3.452

It is recommended that *ElectionIntervalTimer* is set at least to *DSBMDeadIntervalTimer i.e.* 15 seconds [3]. However, from our result this timer may be set to be fired just when *RefreshIntervalTimer* is fired and only the DSBM sends *DSBM_WILLING*. At this time, all DSBM clients know the new elected manager and should transfer their requests to the new DSBM. This makes quick recovery from Failure State and *RSVP PATH* has to be updated to insert the new DSBM instead of the failed one.

We make the following suggestion to quicken the recovery from a failure. At a start of election procedure, all stations generate their candidacy and send it through the LAN segment. This leads to a burst of *DSBM_WILLING* messages. In fact, at every *DSBM_WILLING* message reception, the station must send back a new *DSBM_WILLING* frame if it finds itself better that the received candidate. With the randomness of accessing media for a shared topology, the burst size may cause too much collision. We notice that the messages will be received in random order, which could cause more *DSBM_WILLING* generation. Then, once an SBM client sends its candidacy for the first time, there is no need to re-send its candidacy whenever it receives a *DSBM_WILLING* message.

To reduce this problem over a wide-scale topology, we suggest for an SBM station to not send a new *DSBM_WILLING* in every reception of candidacy message. After the first DSBM candidacy message, SBM station should send a new *DSBM_WILLING* message only after *RefreshIntervalTimer* fire or after a number of successive *DSBM_WILLING* message receptions. This would enhance election process by reducing collisions on shared segment and leads to faster recovery in case of DSBM failure.

For switched topology, there is no collision problem, but for large topology it may be useful to reduce the number of *DSBM_WILLING* messages, to not have overload of switch buffers.

5. Conclusion

In this paper, we have presented a performance evaluation framework for IEEE802.3 style network using SBM protocol as manager of LAN resources. We have evaluated the importance of bandwidth reservation and message scheduling algorithms over a Ethernet LAN to give better response time for real-time sources. The second part dealt with the DSBM election algorithm and proposed some enhancement to achieve faster recovery from a failure state of DSBM.
Based on these results, future work is to build a complete framework for integrated service over an Ethernet network. A possible continuation of this work is to build a QoS framework to support Diffserv request over LAN topology and this presents the advantage to not have a centralized manager for resources as in SBM approach.

6. References

[1] R. Yavatkar, D. Hoffman, Y. Bernet, F. Baker, M. Speer, *SBM (Subnet Bandwidth Manager): A Protocol for RSVP-based Admission Control over IEEE 802-style networks* RFC 2814, May 2000

[2] M. Seaman, A. Smith, E. Crawley, J. Wroclawski " *Integrated Service Mappings on IEEE 802 Networks*" May 2000

[3] A. Ghanwani, W. Pace, V. Srinivasan, A. Smith, M. Seaman "*A Framework for Integrated Services Over Shared and Switched IEEE 802 LAN Technologies*" May 2000

[4] Song, Yeqiong, "*Time Constrained Communication Over Switched Ethernet*", 4th IFAC International Conference on Fieldbus Systems and their Applications - FeT'2001. (Nancy, France).

[5] Koubaa, Anis and Song, YeQiong, "*Evaluation de performance d'Ethernet commuté*", In 10th Conference on Real Time and Embedded Systems. (Paris, France). 2002. 9 p.

[6] Aref Jarraya, Yé-Qiong SONG, Anis KOUBAA « *Evaluation de performance des réseaux Ethernet pour les applications temps-réel* » LORIA-TRIO Internal Report Juin 2002

[7] K.Parekh, G.Gallager, *"A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case",* IEEE/ACM Transactions on Networking Vol1, NO3, June1993

[8] R. Braden,L. Zhang,S. Berson,S. Herzog, S. Jamin, *Resource ReSerVation Protocol (RSVP)* 8, September 1997

[9] An Architecture for Differentiated Services

[10] A.Demers, S.Keshavt, S.Shenker, "*Analysis and Simulation of Fair Queuing Algorithm*", Proceeding ACM SigComm 89, pp 1-12

[11] http://www.opnet.com

Deadline First Scheduling in Switched Real-Time Ethernet – Deadline Partitioning Issues and Software Implementation Experiments

Hoai Hoang, Magnus Jonsson, Anders Larsson, Rikard Olsson, and Carl Bergenhem

School of Information Science, Computer and Electrical Engineering, Halmstad University, Halmstad, Sweden, Box 823, S-301 18, Sweden. {Hoai.Hoang, Magnus.Jonsson, Carl.Bergenhem}@ide.hh.se, http://www.hh.se/ide

Abstract

This paper presents work on a switched Ethernet network extended to allow for earliest deadline first (EDF) scheduling. We show by example that asymmetric deadline partitioning between the links of a real-time channel can increase the utilization substantially, still not violating the real-time guarantees. We also report measurements on a software implementation of the switch on an ordinary PC.

1 Introduction

An important trend in the networking community is to involve more switches in the networks (e.g., LAN, Local Area Networks) and a pure switched-based network becomes more and more common. At the same time, the industrial communication community has a strong will to adapt LAN technology (e.g. Ethernet) for use in industrial systems. The involvement of switches does not only increase the performance; the possibility to offer real-time services is also improved. Now when the cost of LAN switches has reached the level where pure switched-based networks have become affordable, the collision possibility in IEEE 802.3 (Ethernet) networks can be eliminated and methods to support real-time services can be implemented in the switches without changing the underlying widespread protocol standard.

Several protocols to support real-time communication over shared-medium Ethernet have been proposed [1] [2] [3]. However, these protocols are either changing the Ethernet standard or do not add guaranteed real-time services. Real-time communication over switched Ethernet has also been proposed (called EtheReal) [4]. The goal of the EtheReal project was to build a scaleable real-time Ethernet switch, which support bit rate reservation and guarantee over a switch without any hardware modification of the end-nodes. Ethereal is throughput oriented which means that there is no or limited support for hard real-time communication, it has no explicit support for periodic traffic so it is not suitable for industrial applications. A review of research on real-time guarantees in packet-switched networks is found in [5].

This paper presents work on a previously proposed switched Ethernet network with support for both bit rate and timing guarantees for periodic traffic [6]. Only a thin layer is needed between the Ethernet protocols and the TCP/IP suite in the end-stations. The switch is responsible for admission control, while both end-stations and the switch have EDF (Earliest Deadline First) scheduling [7]. Internet communication is supported at the same time as nodes connected to the switch can be guaranteed to meet their real-time demands when they communicate with each other. This is highly appreciated by the industry since it makes remote maintenance possible, e.g., software upgrades or error diagnostics.

The rest of the paper is organized as follows. The network architecture and traffic handling are presented in Section 2. In Section 3, asymmetric deadline partitioning is described and exemplified. Experiments with a software implementation of the switch are then presented in Section 4. The paper is concluded in Section 5.

2 Network architecture and traffic handling

We consider an example of a network with a full-duplex switched Ethernet and end-nodes. Both the switch and the nodes have software (RT layer) added to support guarantees for real-time traffic. All nodes are connected to the switch and nodes can communicate with each other over logical real-time channels (RT channels), each being a virtual connection between two nodes in the system.

In our network configuration, both the switch and the end-nodes use the Earliest Deadline First (EDF) algorithm for traffic control. The switch is responsible for admission control, MAC functions, frame buffering and traffic scheduling. The switch periodically sends synchronization frames to the end-nodes, at an interval, T_{cycle}, of ten maximum sized frames, T_{frame}, i.e.,

$$T_{cycle} = 10T_{frame}. \quad (1)$$

In this way, every node has a uniform comprehension about global time, with the resolution of T_{frame}. In this paper, we assume Fast Ethernet (100 Mbit/s) with a maximum frame size of 12 144 bits which, with some extra time for timing

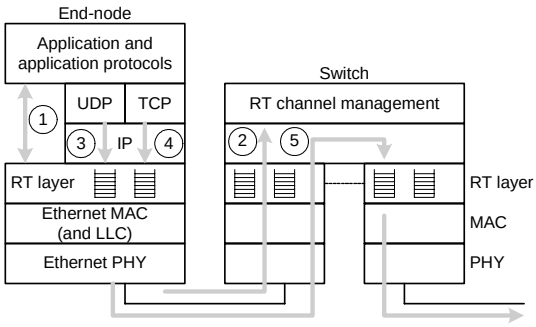


Figure 1: Layers and output queues.

uncertainties and for simplicity, gives $T_{frame} = 125$ µs, which just happens to match the time resolution of many telecommunication systems.

The function of and interaction with the RT layer etc shown in Figure 1 is explained below. When an application wants to setup an RT channel, it interacts directly with the RT layer (1). The RT layer then sends a question to the RT channel management software in the switch (2). Outgoing real-time traffic from the end-node uses UDP and is put in a deadline-sorted queue in the RT layer (3). Outgoing non-real-time traffic from the end-node typically uses TCP and is put in an FCFS-sorted (First Come First Serve) queue in the RT layer (4). In the same way, there are two different output queues for each port on the switch too (5).

An RT channel with index *i* is characterized by:

$$\{T_{period,i}, C_i, T_{deadline,i}\} \quad (2)$$

where $T_{period,i}$ is the period of data, C_i is the amount of data per period, and $T_{deadline,i}$ is the relative deadline used for the end-to-end EDF scheduling. Both $T_{period,i}$, C_i, and $T_{deadline,i}$ are expressed as the number of maximal sized frames, i.e., the number of T_{frame}.

When a node wants to establish an RT channel, it sends a request frame (ReqF) with source and destination node MAC and IP addresses and $\{T_{period,i}, C_i, T_{deadline,i}\}$ to the switch. A connection ID to distinguish between several possible connection requests is also added. When receiving a ReqF, the switch will calculate the feasibility of the traffic schedule between the requesting node and the switch and between the switch and the destination node. The ReqF is then forwarded to the destination node, after adding a network unique ID in the RT channel ID field. The destination node responds with a response frame (ResF) to the switch telling whether the establishment is accepted or not. The switch will then, after taking notation of the response, forward the ResF to the source node.

The RT layer in an end-node prepares outgoing real-time IP datagrams by changing the IP header before letting the Ethernet layers sending it (see Figure 2). The IP source address and the 16 most significant bits of the IP destination address, 48 bits together, are set to the absolute deadline of the frame. A 48 bit absolute deadline with a resolution of $T_{frame} = 125$ µs, gives a "life time" longer than

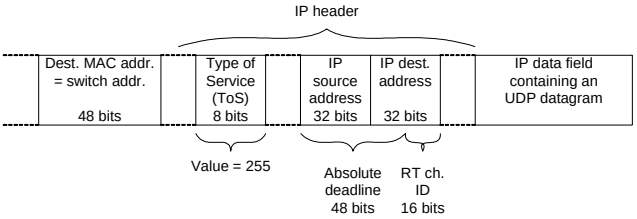


Figure 2: Data frame sent over an RT channel.

one thousand years. The 16 least significant bits of the IP destination are set to the RT channel ID for the RT channel to which the frame belongs. The MAC destination address is set to a special address that all nodes use for real-time traffic, while the Type of Service (ToS) field is always set to value 255.

The switch exchanges the source and destination IP addresses and the MAC destination address of an incoming real-time frame with the correct ones (as stored in the switch when the RT channel was established) for delivery to the final destination.

3 Asymmetric Deadline Partitioning

Below, we will show by example that the possible amount of guaranteed real-time traffic can be increased by partitioning the deadline asymmetrically between the different links crossed by an RT channel. We compare the asymmetric partitioning with a symmetric partitioning for a master-slave situation, i.e., a typical case of real-time traffic pattern in industrial systems.

We assume that master node M_1 is responsible for five slave nodes S_1 to S_5. The master node has one RT channel to each slave node via the switch. The real-time guarantee from M_1 to S_i is uphold by RT channel i, $1 \le i \le 5$. The latency due to synchronization and in-transmission frames [6] is neglected. For the deadline scheduling we assume:

$$T_{deadline,,i} = T_{period,i} = T_{D1,i} + T_{D2,i} \quad (3)$$

for each RT channel i, where $T_{D1,i}$ is the relative deadline for real-time traffic from the master node to the switch and $T_{D2,i,}$ is the relative deadline from the switch to the destination node. In the same way, let us assume that N_1 and N_2 represent the total number of RT channels on the links a specific RT channel crosses, i.e. the load of the links to and from the switch. For simplicity, all channels are assumed having the same characteristics and being unidirectional with the master node as the source node.

With an asymmetric deadline partitioning, $T_{deadline}$ is partitioned so the local deadline for a link of the end-to-end path is weighted according to the load of the link divided by the sum of the loads across the whole path. For our example, this gives:

$$T_{D1,i} = \frac{N_1}{N_1 + N_2} T_{period,i} = \frac{5}{6} T_{period,i} \quad (4)$$

$$T_{D2,i} = \frac{N_2}{N_1 + N_2} T_{period,i} = \frac{1}{6} T_{period,i} \quad (5)$$

This is a simple partitioning method to show the opportunities with deadline partitioning. Our future work includes looking at partitioning method that can handle more complex traffic patterns and dynamic channel setup as the network is designed for.

According to the basic EDF theory [7], the utilization of real-time traffic is defined as

$$U = \sum \frac{C_i}{T_{period,i}}. \quad (6)$$

One is assured that all deadlines are met if the utilization of real-time traffic does not exceed a certain level, $U_{\max}$:

$$U = \sum \frac{C_i}{T_{period,i}} \leq U_{\max} \quad (7)$$

This guarantee holds for deadline scheduling of traffic when the deadline for a specific link is equal to the period multiplied by a constant $k \leq 1$, for all RT channels traversing the link in the same direction. When scheduling a channel with 100 % theoretical utilization, $U_{\max} = k$. For deadline scheduling of traffic with arbitrary deadlines, see [8] or subsequent work (e.g. [9]). We define $U_{\max 1}$ as the maximum utilization for the link from the master node to the switch and $U_{\max 2}$ as the maximum utilization from the switch to the slave node. In the theoretical case $U_{\max}$ is 100 %, but when using the network proposed in this paper the worst-case maximum utilization for the link from the switch and to the slave node is reduced from 100 % to 90 % due to having 10 % bandwidth for the network control. In symmetric deadline partitioning [6], we have $U_{\max 2}$ = 45 % and $U_{\max 1}$ = 50 %. When using asymmetric deadline partitioning with the weights from Equations 4 and 5 (k_1 = 5/6 and k_2 = 1/6, respectively), we get the following maximum utilizations instead:

$$U_{\max 1} = \frac{5}{6} \cdot 100\% = 83\% \quad (8)$$

$$U_{\max 2} = \frac{1}{6} \cdot 90\% = 15\% \quad (9)$$

We denote $C_{\max 1}$ and $C_{\max 2}$ as the maximum capacity (transmission time per period) per channel for the first and the second link traversed by an RT channel, respectively. When assuming the same period, T_{period}, and deadline, $T_{deadline}$, for all RT channels, we have

$$\frac{5C_{\max 1}}{T_{period}} = U_{\max 1} \Rightarrow C_{\max 1} = \frac{U_{\max 1} T_{period}}{5} \quad (10)$$

for the master link and

$$\frac{C_{\max 2}}{T_{period}} = U_{\max 2} \Rightarrow C_{\max 2} = U_{\max 2} T_{period} \quad (11)$$

for a slave link. For example, let us assume that

$$T_{period} = T_{deadline} = 24 \text{ ms} \quad (12)$$

According to Equations 4, 5, 8, and 9, we then have:

$$\begin{aligned} T_{D1} &= 20 \text{ ms} \\ T_{D2} &= 4 \text{ ms} \\ C_{\max 1} &= \frac{U_{\max 1} T_{period}}{5} = 4 \text{ ms} \\ C_{\max 2} &= U_{\max 2} T_{period} = 3.6 \text{ ms} \end{aligned} \quad (13)$$

The second link is the bottleneck because $C_{\max 2} < C_{\max 1}$. With $C = C_{\max 2}$ = 3.6 ms for all RT channels we get a utilization of $U = C / T_{period}$ = 0.15 on each slave link and $U = 5C / T_{period}$ = 0.75 on the master link.

In the asymmetric case, we have 75 % utilization on the master link compared with 50 % in the symmetric case. We still guarantee worst-case delay for real-time traffic. The analysis given above holds for the opposite direction (from each slave and to the master) and for other, not overlapping, master slave groups in the network too.

4 Software Implementation

We have implemented the switch using a PC with a 200 MHz Pentium processor, some network interface cards and LINUX 2.4.2. The two most important parts in the switch regarding the real-time communication are the RT-layer and the RT channel management (see Figure 1). The RT channel management is responsible for approving RT channels requested by nodes in the system. Information about the approved RT channels is made available to the RT layer, which handles the actual scheduling and forwarding of data frames. If the load on the switch becomes too high it simply discards non-RT frames.

The tests were performed in a network with the switch and three nodes, two sending and one receiving. All nodes were equipped with 100 Mbit/s full-duplex Ethernet cards. The PC acting as switch operates at 200 MHz. In the first test, the frame size, N_{byte}, is set to 1 000 bytes, including headers. In the receiving node, a timer, $t_{measure}$, starts at the arrival of the first RT frame and is stopped when the last frame is detected. During this time all received frames are registered either as RT frames or non-RT frames. The number of registered RT frames and non-RT frames are denoted as N_{RT} and N_{NRT}, respectively. The corresponding data rates for received data, R_{RT} and R_{NRT}, are calculated as:

$$R_{RT} = \frac{8 N_{Byte} N_{RT}}{t_{measure}} \quad (14)$$

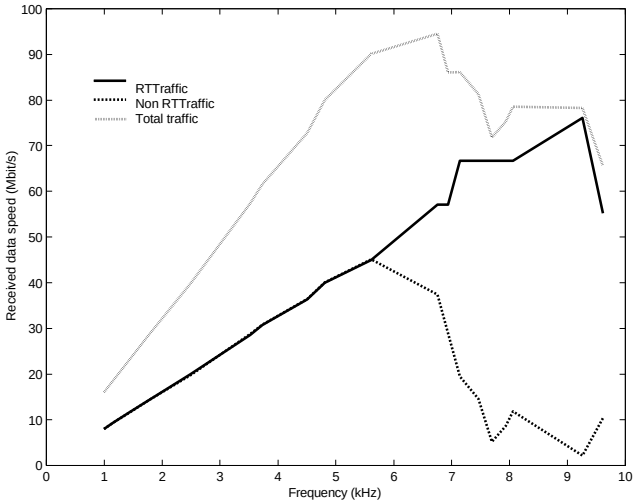


Figure 3: Test results.

$$R_{NRT} = \frac{8N_{Byte}N_{NRT}}{t_{measure}} \quad (15)$$

The measured data rates in the receiving node are plotted against the frequency of the injected periodic traffic (see Figure 3). In this test, both RT traffic and non-RT traffic were injected with the frequency indicated in the figure. The result shows that the switch prioritizes the RT frames. It also shows that the breakpoint, where the switch begins to discard non-RT frames, is when the total traffic load reaches 96 Mbit/s, i.e., when the total traffic-load approaches the maximum capacity of the switch, the switch starts to discard non-RT traffic. As the traffic load increases, the RT-Switch is finally forced to discard RT traffic as well. This happens when the period of the traffic from the sending application is about 0.1 ms. This situation can only arise when the system runs without any channel management.

Additional tests where made including the two extremes: (*i*) Only maximum sized frames (1518 bytes) are transmitted. This gives a maximum throughput for the switch, measured to 96 Mbit/s, which is near wire speed. (*ii*) Only minimum sized frames (64 bytes) are transmitted. This gives the lowest throughput, measured to 18 Mbit/s. This case also gives us a maximum switching capacity of 14 400 frames/s.

The tests that have been performed shows that it is possible to build an Ethernet switch with deadline scheduling by using only standard components. This gives an indication on the possibility of implementing real-time capabilities in an Ethernet network. One can get a good feeling from the measurements about the amount of real-time traffic and number of ports that can be supported if, for example, a processor should assist a switch chip by handling deadline scheduling in software.

5 Conclusions

In this paper, we have presented an Ethernet network with support for real-time traffic by deadline scheduling. We have showed by example that the performance can benefit a lot from asymmetric deadline scheduling. An increase in utilization from 50 % to 75 % on the outgoing link from the master node (the bottleneck) is observed, still not violating the real-time guarantees.

From a software-implemented switch, we have showed experimental results. The measurements showed that the switch bottlenecks are 96 Mbit/s (measured for maximum-sized frames) and 14 400 frames/s (measured for minimum-sized frames). From these measurements one can get a good feeling of the amount of real-time traffic and number of ports that can be supported by an Ethernet switch that is fully or partly implemented in software.

References

[1] C. Venkatramani and T. S. Chiueh, "Supporting real-time traffic on Ethernet," *Proc. 15th IEEE Real-Time Systems Symposium (RTSS'94)*, pp. 282-286, 1994.

[2] D. W. Pritty, J. R. Malone, D. N. Smeed, S. K. Banerjee, and N. L. Lawrie, "A real-time upgrade for Ethernet based factory networking", *Proc. IEEE IECON'95*, vol. 2 , 1995.

[3] S.-K. Kweon, K. G. Shin, and G. Workman, "Achieving real-time communication over Ethernet with adaptive traffic smoothing," *Proc. 6th IEEE Real-Time Technology and Applications Symposium (RTAS'2000)*, Washington, D.C., USA, 31 May - 2 June 2000, pp. 90-100.

[4] S. Varadarajan and T. Chiueh, "EtheReal: A host-transparent real-time Fast Ethernet switch," *Proc. ICNP*, Oct. 1998.

[5] H. Zhang, "Service disciplines for guaranteed performance service in packet switching network," *Proc. of the IEEE*, vol. 83, no. 10, Oct. 1995.

[6] H. Hoang, M. Jonsson, U. Hagström, and A. Kallerdahl, "Switched real-time Ethernet with earliest deadline first scheduling - protocols and traffic handling", *Proc. Workshop on Parallel and Distributed Real-Time Systems (WPDRTS'2002) in conjunction with International Parallel and Distributed Processing Symposium (IPDPS'02)*, Fort Lauderdale, FL, USA, April 15-16, 2002.

[7] C. L. Liu and J. W. Layland, "Scheduling algorithms for multipprogramming in hard real-time traffic environment", *Journal of the Association for Computing Machinery*, vol. 20, no. 1, Jan. 1973.

[8] D. Ferrari and D. C. Verma, "A scheme for real-time channel establishement in wide-area networks," *IEEE Journal of Selected Areas in Communications*, vol. 8, no. 3, pp. 368-379, Apr. 1990.

[9] Q. Zheng and K. G. Shin, "On the ability of establishing real-time channels in point-to-point packet-switched networks," *IEEE Transactions on Communications*, vol. 42, no. 2/3/4, pp. 1096-1105, Feb./Mar./Apr. 1994.

Designing, Modelling and Evaluating Switched Ethernet Networks in Factory Communication Systems

N. Krommenacker, J.P. Georges, E. Rondeau, T. Divoux
CRAN – CNRS UMR 7039, Université Henri Poincaré - Nancy 1
BP 239 - 54506 Vandoeuvre lès Nancy - FRANCE
E-mail : <nicolas.krommenacker@cran.uhp-nancy.fr

Abstract

The Ethernet network is increasingly used for the industrial communications which are strongly time-constrained. This paper presents both a method (based on the genetic algorithms) to design switched Ethernet architectures and a method (based on the network calculus) to evaluate these network architectures in term of maximum end-to-end delay.

1. Introduction

The industrial communications are currently based on specific networks called fieldbuses such as FIP, Profibus, CAN. They interconnect programmable controllers, CNC, robots, … to exchange technical data for monitoring, controlling, and synchronising industrial processes. Their main characteristic is to ensure that the end-to-end delays of messages remain limited, compared with the time-cycle of the applications. Thus, these networks are deterministic and some protocols satisfy the integrity constraints of the information. In opposition, the Ethernet networks based on the CSMA/CD protocol are increasingly used to interconnect industrial devices. Some applications confirm this evolution in different industrial areas : cars (Jaguar), pharmaceutics (Boehringer Ingelheim),… Moreover, different organisations such as the Industrial Automation Networking Alliance (IAONA), the Industrial Ethernet Association (IEA) promote Ethernet as "the standard in the industrial environment". Finally several research projects such as CIDER (Communication Infrastructure for Dependable Evolvable Real-Time Systems) [1] study the use of Ethernet as an enabling technology for future dependable real-time systems.

In the first part of this paper, we propose to design a switched Ethernet architecture which satisfies the time requirements. Our objective is to optimise the network organisation at the physical layer level. To define efficient topologies, we use graph partitioning techniques in order to distribute industrial devices on the different Ethernet switches. The graph partitioning is an NP-complete problem for which several heuristics have been developed. The size of the search space for an arbitrary problem instance is defined by the number of nodes and partitions. We can distinguish two types of approaches. The constructive methods elaborate a solution *ex nihilo* while the improvement methods try to improve a given solution with the help of a heuristic. In this study, we suggest the use of such a heuristic : the Genetic Algorithms.

In the second part, we describe an approach to evaluate the network organisation obtained from the network design step based on genetic algorithms. In the literature, several works use a stochastic modelling in order to evaluate the switched Ethernet performances. Nevertheless, these surveys study the communication system with input services such as Bernoulli processes, Poisson processes, … which are not representative to the messages sent by the industrial devices. Basically, a programmable controller periodically sends data on the network with critical temporal constraints and also sends aperiodic messages. Thus we propose to model these communications by using a strict arrival curve and in particular by using the network calculus. The network calculus introduced by R. Cruz [2][3] only assumes that the number of bytes sent on the network links does not exceed an arrival curve service (traditionally, a leaky bucket) which can represent both the periodic and the aperiodic traffic. The other interest to use the network calculus is to be able to model the Ethernet switches and their interconnection by assembling basic components such as multiplexer, demultiplexer, buffer, … whose temporal properties were given by R. Cruz. The network calculus enables then to determine the packet maximum end-to-end delay or jitters from exchange matrices which model the traffic between the industrial devices.

Finally, we study in this paper a communication scenario between programmable controllers in order to illustrate the interests of our proposals.

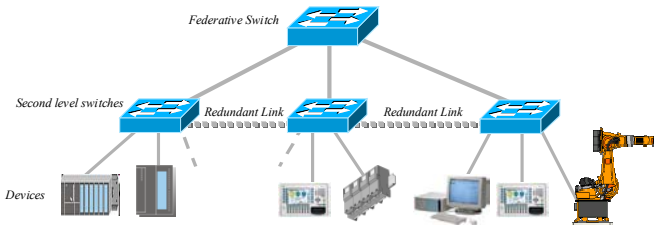


Figure 1. Network Topology

2. Designing switched Ethernet networks

2.1. Introduction

It exists two main kinds of topologies to connect the industrial devices on switched Ethernet networks. The most common are the hierarchical and the linear topologies. In this paper, we analyse network architectures based on these two topologies (figure 1). The hierarchical topology is activated in the normal functioning mode and when it is out of order, the linear topology is activated in order to guarantee the network connectivity.

The general method to design local area network always consists both in concentrating the traffic inside each network segment and in balancing the load between the different segments. For this, the exchanges are represented on a graph where the vertices correspond to the manufacturing devices and the weighted edges model the quantity of communication between them. Then, the network segmentation step is achieved by using graph partitioning techniques. The originality of our approach, in opposition to previous works [4][5], is to define a fitness function which is very efficient since in a quick time we can obtain a good network architecture.

We present in the next sections, the main principles to design network architecture by using the genetic algorithms[6].

2.2. Coding structure and Parameters

For the graph partitioning problem, several encoding methods exist. The method that is the most often used is the N-string representation. Each chromosome corresponds to a vector in which the i^{th} element of an individual is j if the j^{th} vertex of the graph is allocated to the partition labelled j. There is as many elements in the vector as vertices in the graph. The multiway partitioning problem uses the k-ary alphabet [0,1,…,k-1]. For instance, the string [001221120] represents the mapping that assigns vertices 1, 2, 9 to partition 0, vertices 3, 6, 7 to partition 1 and vertices 4, 5, 8 to partition 2. The figure 2 illustrates the encoding of this solution and the decoding of this solution as a hierarchical topology with redundancy.

[6] defined conventional GA parameters such as population size (pop_{size}), crossover (p_c) and mutation (p_m) probability. These parameters influence the performance of the algorithm. For example, the size of the initial population influences on the quality of the final solution to the detriment of the execution time. For this study, the good results were found with the following parameters : pop_{size}=15, p_c=0.2 and max_{gen}=500.

To start the GA, the population has to be initialised. The common method is to create solutions at random. Nevertheless, the random method must be controlled in order to avoid to create absurd chromosomes such as [002220020] or [11111111] where the final expected number of groups is not respected. For the selection step, the "roulette wheel" approach has been chosen. This is a very common parent selection method where each chromosome selected as a parent has a probability that is proportional to its fitness. Moreover, a elitist strategy is used to keep the best individuals from the current generation to the next generation.

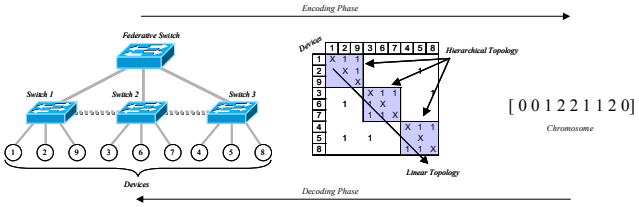


Figure 2. Encoding Network Topology

2.2. Crossover and Mutation operators

In GA, the crossover and mutation operators are the two most important space exploration operators. A crossover operator creates a new offspring chromosome by combining parts of the two parent chromosomes. A two-point crossover is employed in our algorithm. Two cut points are randomly selected which are the same on both parent chromosomes. Each chromosome is also divided into three disjoint parts. A new chromosome is formed randomly copying a part from one parent to the corresponding part of the offspring (see figure 3).

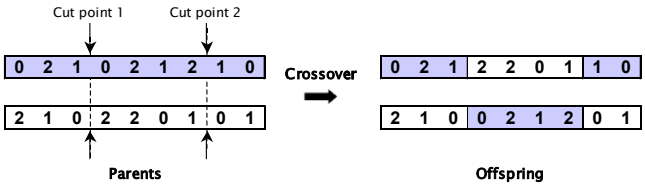


Figure 3. Two-point crossover operator

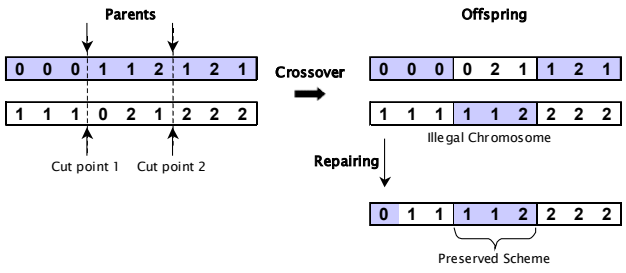


Figure 4. Repairing step

The crossover operator can generate illegal chromosomes as in the initialisation step. In the figure 4, an offspring violated the number of group. We propose a repairing procedure which consists in checking the group

number constraint. If a chromosome violates this constraint, it is repaired with permutations from parent chromosomes. The permutations are applied out of the cut-points in order to preserve the scheme of parent chromosome.

The mutation operator enables to introduce unexplored search space to the population. The method that is the most often used is the bit-flop mutation. However, a swap mutation is used in order to control the constraints of the group number.

2.3. Fitness function

To determine the quality of a solution, we have used two metrics. The first one is the edge-cut criterion which represents the volume of the inter-group exchanges. It is the sum of all the edges which connect a vertex in one partition to a vertex in an another partition. The second one is the set size criterion which measures the balance between groups. [7] defined a grouping efficacy measure that minimises the number of outside elements of the partitions (exceptional elements) and void elements inside the partitions. For each solution, a fitness value is calculated with the following fitness function :

$$f(e, e_V, e_0) = \frac{e - e_0}{e + e_V}$$

with :

- e : sum of elements inside matrix
- e_0 : sum of elements outside partitions
- e_v : sum of null elements inside partitions

Thus, the objective is to identify a network architecture which maximises the fitness function f.

3. Modelling switched Ethernet networks

In this section, we propose to model an Ethernet switch. [8] and [9] decompose the switching architecture in three main components :

- the queuing model refers to the buffering and the congestion mechanisms located in the switch. In our modelling, we consider an organisation based on the shared memory queuing.
- the switching implementation refers to the decision making process within the switch (how and where a switching decision is made). We select a centralised switching implementation.
- And the switching fabric is the path that data takes to move from one port to another. From previous choices, the shared-memory architecture is chosen.

The figure 5 lists the different components retained in this paper to model a switching architecture. It is constituted of a sequence of three components : one multiplexer, one FIFO queue and one demultiplexer. This model could be applied to different industry products, like the Cisco Catalyst 2009XL. The interconnection of the switches enabling to build the network architecture is achieved by using links which are modelled by buffers. The link transmission mode in this work is supposed to be the full-duplex mode in order to avoid collision problems introduced in the half-duplex mode.

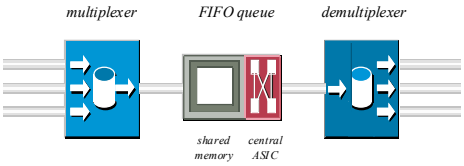


Figure 5. Switch modelling

4. Evaluating switched Ethernet networks

4.1. Traffic modelling

In the industrial context, there are two kinds of communications between the devices : the periodic and aperiodic messages. To represent the traffic, we use the leaky bucket controller concept (definition 1.3.2 in [10]). It imposes the traffic generation to be bounded by an affine function named leaky bucket, noted b(t), in which a variable burst value σ is associated to a constant rate ρ. [2] proposes also to associate a burstiness constraint based on the leaky bucket to each stream and in which : $b(t)=\sigma+\rho t$

$$\text{And } R(t)<b(t) \Leftrightarrow \int_{t=x}^{y} R(t)dt < \sigma + \rho(y - x)$$

Where :

- R(t) represents the instantaneous rate of all traffics merged (periodic and aperiodic) from the stream,
- σ is the maximum amount of traffic that can arrive in a burst,
- ρ is an upper bound on the long-term average rate of the traffic flow.

This peakedness characterisation of the traffic presents two advantages. First, the leaky bucket gives a deterministic representation since we are sure that the number of arrived bytes from a stream i at a time t is bounded by $b_i(t)$. Secondly, as the leaky bucket form is an affine function, it is possible to merge periodic and aperiodic data. Since the leaky bucket function is affine, the number of data transmitted by the programmable controller is bounded by : $b_{total}(t) = b_{periodic}(t) + b_{aperiodic}(t)$.

4.2. End-to-end delay

We have defined a general model of a switched Ethernet architecture which is based on four components (multiplexer, FIFO queue, demultiplexer and buffer). The network calculus developed by R. Cruz enables to obtain the upper bounded of end-to-end delay (with a lucky

bucket modelling) for each component. To determine the maximum end-to-end delay in a complete switched communication system, we must add the different maximum end-to-end delays. The general formula which we obtain is :

$$\overline{D_{switch,i}} = \alpha\left[\sigma_1 + \sigma_2 + ... + \beta_{\sigma_1} + ... + \Lambda\sigma_K + ... + \sigma_M - \delta\right]$$

and is explained in [11]

5. Example

To illustrate the behaviour of the network design algorithm proposed in this paper, we study a traffic matrix which collects the information exchanges between 40 programmable controllers (Figure 6). The whole messages is represented by a binary matrix and each n in the matrix corresponds to one packet sent every 5 milliseconds. The size of each packet is equal to 64 bytes (minimal size of a Ethernet packet). We use 12-ports Ethernet switches with full-duplex links at 10Mb/s. The objective is to find a good distribution of the programmable controllers on five switches.

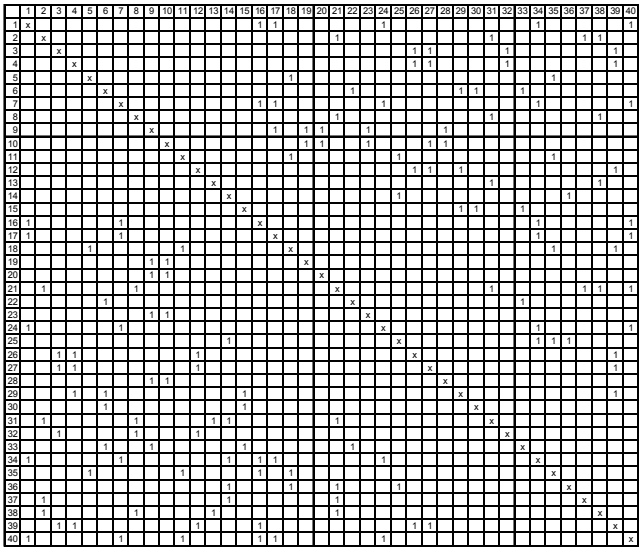


Figure 6. Exchange matrix

Before analysing the topology obtained from our algorithm, we have searched a bad partitioning which is composed of five groups constituted of the following programmable controllers : 1-9, 10-17, 18-26, 27-33 and 34-40. In this solution, all the communication are inter-group communications. The total traffic is outside partitions and the fitness function is also null. The mapping of this solution corresponds to an architecture with a federative switch which supports all the traffic defined in the matrix. The genetic algorithm proposed in this paper is implemented in Java language. Since GA is a stochastic search algorithm, it is performed 20 times. The best partition obtained is : {1,7,16,17,24,34,40}, {3,4,12,15,26,27,29,30,32,39}, {5,11,14,18,25,35,36}, {2,8,13,21,31,37,38}, {6,9,10,19,20,22,23,28,33}

From the bad and good organisations, we can compare the end-to-end delays by using the network calculus. We find for the bad solution a maximum end-to-end delay equal to 914 µs and the means of all maximum end-to-end delays is equal to 819 µs. The same operations applied to the good solution give respectively 880 µs and 500 µs. We observe a significant difference since between the bad and the good organisations the means of all maximum end-to-end delays are divided by two

6. Conclusion

This paper proposes a general method to design and evaluate a switched Ethernet architecture for real-time applications. The next steps of this work are to implement in the switch model, some mechanisms of Quality Of service such as priority as defined in 802.1p and to introduce directly in the fitness function the maximum end-to-end delay formula.

7. References

[1] M. Alves, E. Tovar, G. Fohler, G. Buttazo "CIDER – envisaging a cots communication infrastructure for evolutionnary dependable real-time", *12th Euromicro conference on Real-Time Systems*, Stockolm, pp.19-22.

[2] R. Cruz, "A calculus for network delay, Part I : Network elements in isolation " *IEEE Transactions on Information Theory.* Vol. 37, pp. 114-131, 1991

[3] R. Cruz, "A calculus for network delay, Part II : Network analysis" *IEEE Transactions on Information Theory.* Vol. 37, pp. 132-141, 1991

[4] R. Elbaum, R. Sidi "Topological design of local area networks using genetic algorithms", *IEEE/ACM Transactions on Networking*, Vol. 4, n°5, pp.766-778, 1996

[5] M. Nusekabel "Switched network partitioning using Tabu search", *Thesis for Master of Science in Computer Science*, University of South Florida, 1997.

[6] D.E. Goldberg "Genetic algorithms in search optimization, and machine learning". *Addison Wesley*, 1989.

[7] C.S. Kumar, M.P. Chandrasekharan "Grouping efficacy : a quantitative criterion for goodness of block diagonal forms of binary matrices in group technology" *Journal of Production Research*, Vol. 28, n°2, pp. 233-243, 1990.

[8] M. Philipps "A guide to LAN switch architectures and performance" Packet CISCO Systems users magazine, pp.54-57, 1999

[9] R. Seifert "The switch book" *John Wiley and sons*, New-york, 2000

[10] J.Y. Le Boudec, P. Thirian "Network calculus" *Springer Verlag*, 1990.

[11] J.P. Georges, E. Rondeau, T. Divoux "Using the network calculus to evaluate the performances of a switched Ethernet network in an industrial context" *Internal report*, CRAN, 2002.

Real Time on Ethernet using off-the-shelf Hardware

Jork Löser Hermann Härtig

TU Dresden, Germany

Abstract

Switched networks increasingly become commodity, replacing shared bus networks in LANs. Switched networks support simultaneous access using dedicated channels per attached node and reduce frame drops using buffers. We use these two properties to achieve lossless real-time data transfer at the network level. In this paper, we describe the model and our implementation in a real-time Operating System. This entirely software-based solution provides application-to-application real-time communication on standard hardware using UDP/IP as transport level protocol.

1 Motivation

To provide real-time communication we have to 1) guarantee timely delivery of data frames at the network level, to 2) prevent data loss at the network level and to 3) provide the real-time guarantees to user applications using an appropriate Operating System.

After the availability of ATM networks, that provide real-time transfer at the network level, it took quite some time until Operating Systems were able to provide this real-time transfer to the application level [2]. We show in this paper, that recent developments in the Ethernet technology allow us to use the popular and cheap Ethernet to transfer real-time data efficiently as well. Again it seems, that this potential is not used by OS implementations yet, regardless of significant advantages over ATM: Due to the high costs inherent to the complex ATM technology, ATM did not become as widely used as expected. This urges investigating other, more common network technologies, e.g., Ethernet. To provide real-time data transfer to the application level efficiently, we developed a real-time network stack running on our real-time operating system DROPS.

2 Hardware issues

We focus our work on the Ethernet technology, **the** commodity network for decades. Within the original bus-based Ethernet, *collisions* appear as a result of the the CSMA/CD technology. These collisions lead to automatic retransmissions, which in turn prevent to give tight bounds for the transfer time of data.

With *switched* Ethernet the bus-based data exchange turned into a star-based one: Every node has a pair of exclusive channels to transmit to and receive data from a central switch. The switch receives and forwards the data to the according destination. CSMA/CD is not used and the absence of collisions results in upper bounds for data transfer times.

Still, in cases of high load switches must drop frames. To prevent this dropping, let us investigate what high load means, i.e., when a switch actually drops frames. For the sake of simplicity, we concentrate on an output-buffered switch. Figure 1 shows a typical switch with receive channels (rx channels), control logic, buffer space and queued transmit channels (tx channels).

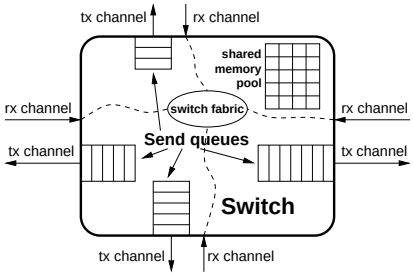


Figure 1: Queuing inside an output-buffered Switch. If queueing a frame is necessary, memory is allocated from a shared memory pool and assigned to the corresponding queue.

If a frame arrives for an output channel where the control logic is forwarding another frame to, the frame is queued. If all memory inside the switch is allocated for queued frames, the current frame must be dropped.

Hence, all that needs to be done to prevent dropping

of frames is to avoid the switching memory of being exhausted, i.e., to bound the output queues in length. This requires two conditions to be met: Firstly, the accumulated average rate of incoming traffic designated for one particular transmit channel must not exceed the traffic rate of the transmit channel. Secondly, the amount of data arriving in a particular time interval must be bound.

Formally, let B be the bandwidth of a channel, measured in number of maximum sized frames per second. Let N be the number of nodes sending to the according output channel, and let b_i the bandwidth (in frames) node i is allowed to send with. Let further M_i be the amount of memory (in frames) the according output queue in the switch is allowed to occupy on behalf of node i.

Using a (Λ_i/E_i) leaky-bucket traffic shaper [4] at the transmitter of each network node results in the desired bounding of the queue lengths. We set $\Lambda_i = b_i$ (the average bandwidth) and $E_i = M_i$ (the maximum burst size). The parameters b_i and M_i are determined based on user-requests for bandwidth and validated by an appropriate software-based reservation mechanism.

When determining values for M_i, minimizing message delays caused by queueing in the switch conflicts with minimizing the CPU consumption in the traffic shaper: Longer maximum queue length increase the maximum delays, smaller bucket sizes may require the traffic shapers to runs more often. When the bucket sizes of a connection are set proportionally to the bandwidth of that connection, the time to refill the bucket becomes a constant. With Fast Ethernet and a switch buffer capacity of 512KByte this refill time has an upper bound of 41.9 ms. Todays workstations are fast enough to run leaky bucket shapers with a refill time of 1 ms.

Note, that the proposed traffic shaping does not require to modify the switch or the node hardware. It can entirely be executed in software at the driver level. In contrast to token-based or time-slot mechanisms this scheme has the advantage that all nodes can perform their send operations independently and specifically unsynchronized after a connection is established. Obviously, the traffic shaping scheme can be extended to multiple connections at each node. Each connection has its own traffic shaper with an own set of parameters.

3 Implementation Issues

Our system is built on the Dresden Realtime OPerating System DROPS [1], a micro-kernel based system.

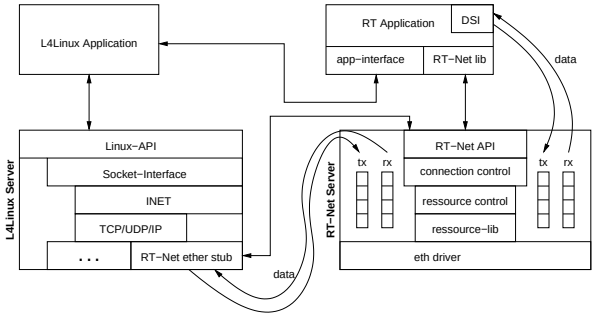


Figure 2: The node architecture.

DROPS runs real-time applications, which reserve the resources they need for proper operation. Remaining resources, including CPU cycles, memory and network bandwidth, can be consumed by best-effort applications. One of these best-effort applications is L^{4}Linux, a server offering the Linux kernel API to execute Linux applications. Consequently, L^{4}Linux utilizes a stub driver to access the network using our network stack.

Figure 2 shows the application model of our approach. An *RT-Net Server* directly interacts with the network interface card (NIC). The server shapes the outgoing traffic according to prior reservation and polices incoming traffic to avoid overload situations. It offers connection-oriented packet-based interfaces to its clients. This allows accounting of transmit traffic and early demultiplexing of received traffic, for real-time clients as well as for best-effort clients.

Best-effort clients normally implement IP-stacks, and hence transfer data-link layer frames to the RT-Net Server. In contrast to this, real-time clients are user applications operating at the transport layer. We use UDP/IP to transfer real-time data, and hence real-time connections are UDP/IP connections with fixed IP addresses and fixed UDP ports. The UDP protocol handling is done entirely at the RT-Net server. For the data exchange between the RT-Net server and the clients a zero-copy IPC protocol [3] is used. However, as we use standard NICs, receiving data requires one copy operation within the server.

3.1 Receiving Process

The receiving process runs in its own thread at interrupt priority inside the RT-Net server. Immediately after a frame is received from the NIC, early demultiplexing is used to find the appropriate receive-connection for that frame. To find a connection, the demux al-

gorithm checks the layer-3 protocol id (IP), the IP-protocol (UDP), the destination address and the destination port of a frame. This requires 2 compare-operations per frame and two additional compares for each real-time client and frame. If no real-time receiver is found, the frame is processed as a best-effort frame.

3.2 Sending Process

Contrary to the receiving process, the sending process is multithreaded, utilizing one thread per connection. Each thread waits for its client to provide a packet. If a packet is obtained on a real-time connection, it is encapsulated using appropriate UDP/IP headers. Note that this is a very fast operation, because the header information is mostly static for the packets of one connection. The connection is traffic-shaped then using a leaky bucket algorithm. Immediately after this, the packet is enqueued at the NIC.

Zero-copying is provided for both real-time and best-effort connections. For real-time connections, the RT-Net Server manages the shared memory used for the connection, which is a physically contiguous piece of memory. Hence, it can calculate the physical addresses of the data therein without effort, which it passes to the NIC. The prepended UDP/IP-headers are passed to the NIC using scatter/gather-techniques. Contrary to real-time clients, best-effort clients are trusted by the RT-Net Server. They pass physical addresses of their data to be send, which is directly passed to the NIC. Hence, the RT-Net Server has no need to access (and copy) best-effort send data.

Prior to establishing a connection at the RT-Net Server, a bandwidth reservation for the intended connection is required. Therefore, a management instance on a network-connected host is contacted, the *bandwidth manager*. The bandwidth manager assigns some amount of the switch buffer memory to each connection and ensures the switch memory not to be overbooked. It also ensures that the bandwidth reservations do not exceed the channel capacities.

3.3 Best-Effort Send Traffic

A problem specific to best-effort traffic is its sporadic burstiness. In contrast to real-time traffic, which uses bandwidth reservations based on prior analysis, best-effort traffic tends to be unpredictable. Moreover, best-effort traffic should utilize all remaining bandwidth, which is not used by real-time traffic. And last but not least, multiple best-effort senders in a network should be able to share the unoccupied bandwidth. Therefore, reserving a fixed bandwidth for each best-effort connection is not an option.

Instead, we reserve only a small amount of bandwidth for every best-effort send connection (i.e., an IP stack normally). If the best-effort sender realizes it needs a higher bandwidth, it tries to make an additional *one-shot reservation*. This one-shot reservation is valid only for a short period of time immediately after the reservation. During this time, the sender can transmit its data. If the time is over, and the sender still has to send a lot of data, it tries to make a reservation again.

When shaping the outgoing traffic at a node, we do not analyze where a best-effort traffic frame is sent to, currently. Therefore, the bandwidth manager takes care of all output queues of the switch when handling best-effort reservation requests.

3.4 Initial Sending

To cope with the problem of establishing the first connection of a node (which is used to establish further connections), we pragmatically reserve a very small amount of bandwidth for every node attached to the network.

An alternative we have in mind is to use traffic prioritizing for the case that the used switch honors priority tagging. Analogously to ATM, all traffic that is sent conforming to a reservation, is marked with a high priority. Other traffic is send with a low priority. While traffic shaping is still required for all reserved connections, prioritizing has the following advantages: The initial traffic to establish the first connection can be sent with a low priority. Hence, we do not need to reserve that small amount of bandwidth for every potentially sending node in the network. Also, the best-effort traffic that exceeds the best-effort reservation could be sent with a low priority. In the case that bandwidth is left, the best-effort traffic passes the switch successfully. In the other case, it is discarded. Unfortunately, TCP/IP performance suffers dramatically from frequently dropped frames. It is on our agenda to look for a solution to this, currently we do not use traffic prioritizing.

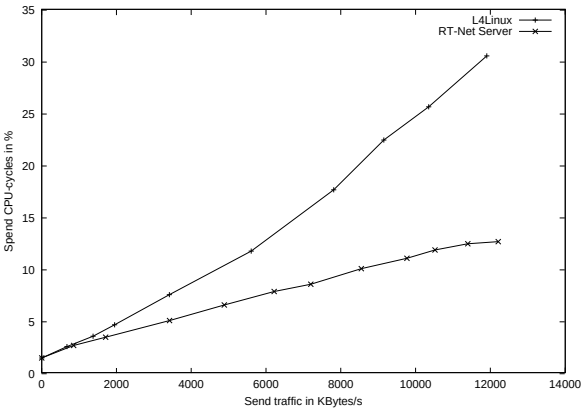


Figure 3: CPU cycles spent for sending data.

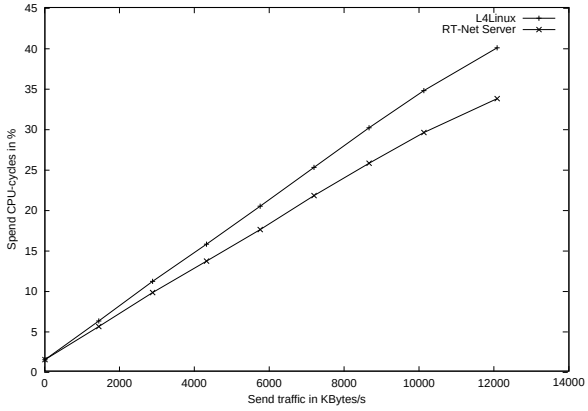


Figure 4: Used CPU cycles when receiving data.

4 Measurements

We measured the CPU time spend by DROPS and our network stack depending on the network load of real-time applications. We set up a send-connection which sends data with a bandwidth according to given reservations. During the experiment, we vary the reservation and hence the bandwidth. The traffic shaper uses a shaping interval of 1 ms all the time: If the bucket becomes empty, it delays frames for at least 1 ms to allow the bucket to fill. To investigate what our scheme for real-time on Ethernet actually costs, we compare the performance of our real-time stack with that of the original L^{4}Linux implementation. For that, we used an Linux application sending UDP datagrams. We sent bursts of different sizes and used `usleep` system-calls of 10 ms between these bursts. The achieved bandwidth was calculated of the amount of data being sent and the elapsed time. The spend CPU cycles in the real-time case and the L^{4}Linux case was measured with a low-priority (i.e., niced) process consuming spare CPU cycles and averaged over a 10 seconds: The less CPU time it got, the more CPU time was spend in DROPS and the network stack, resp. L^{4}Linux.

All experiments were done on an Intel Celeron Processor with 900MHz and 128KByte second level cache. The Fast Ethernet NIC uses shared ring-buffers to communicate with the host. The host writes send or receive descriptors into these rings and the NIC uses PCI-DMA to transfer the data of a frame. Figure 3 shows the time spent for sending data when using the traffic-shaping real-time stack resp. when using original L4Linux (100% correspond to 900 Mio cycles).

To measure the impact of receiving data, we used a similar setup. Here we offered a load to the host, which was consumed either by an real-time application, or, for the original L4Linux, by an user-process receiving the data. Figure 4 indicates the CPU cycles used for these cases.

As you can see, our real-time stack consumes less CPU-cycles than the L^{4}Linux IP-stack implementation. This is mainly due to the small overhead our network stack imposes for data transfer in contrast to L^{4}Linux, which copies the data between the user application and the kernel and executes more code in its network stack. The performance difference for the receive direction is mainly due to the early demultiplexing, which saves a lot of checks and queuing operations compared to the original L^{4}Linux.

References

[1] Dresden Realtime OPerating System. Project overview: `http://os.inf.tu-dresden.de/drops/`.

[2] Martin Borriss and Hermann Härtig. Design and implementation of a real-time ATM-based protocol server. In *19th IEEE Real-Time Systems Symposium (RTSS)*, Madrid, Spain, December 1998.

[3] Jork Löser, Lars Reuther, and Hermann Härtig. Position summary: A streaming interface for real-time interprocess communication. In *8th Workshop on Hot Topics in Operating Systems (HotOS)*, Elmau, Germany, May 2001. A comprehensive Tech Report is available from URL: `http://os.inf.tu-dresden.de/~jork/dsi_tech_200108.ps`.

[4] J. S. Turner. New Directions in Communications (or Which Way to the Information Age?). *IEEE Comm. Magazine*, 24(10):pp. 8–15, October 1986.

Bluetooth - One of the Best WPAN Solutions for Bridging PAN and Wider Networks?

Tibor Dulai, *dtibor@vekoll.vein.hu*

Anna Harmatné Medve, *medve@almos.vein.hu*

University of Veszprém, Hungary
Institute of Information Technology and Electrical Engineering
Department of Information Systems

Abstract

In this paper we present the advantages of Bluetooth to other WPAN technologies and try to answer the question: why is Bluetooth a promising WPAN technology and bridge to wider networks. Finally we give a method to design protocols which make the wide usability possible with the help of patterns.

Keywords: Bluetooth, WPAN, protocol engineering, SDL, design pattern

1. Introduction

Nowadays it is important to get the important information from everywhere as soon as possible. The devices make it possible have to be designed to be handheld and mobile. To access the information these WPAN equipments have to connect to other devices or networks. One of the technologies developed for wireless short range communication is Bluetooth. In the following part we will see, why is Bluetooth a very promising technology for this kind of communications.

2. Why is Bluetooth so suitable for WPAN ?

Communicate with help of RF signals has a lot of advantages. We do not have to point our devices at each other because the propagation of radio waves is independent of direction, even more RF signals passes over non metal objects. That's why this part of electromagnetic spectrum is ideal for mobile communication. We can take our mobile phone or PDA into other room without breaking the connection. It is one of the most important properties of WPAN devices. Several different technologies have been developed for short range RF communication, for example Home RF, IEEE 802.11 for WLAN and Bluetooth. Each of them has its special environment and application optimized for. Home RF, IEEE 802.11b and Bluetooth work in the 2.4 GHz ISM band, that's why it doesn't need any radio license to use them. It causes a big problem: this radio band is full of signals which can generate interference between communicating devices. Bluetooth applies a quick frequency hopping scheme to avoid interference by using FHSS with a nominal 1600 hops/sec speed.

The ideal WPAN devices have another important attribute: they are small enough to fit in a pocket, they minimize power consumption and are as cheap as possible. Bluetooth satisfies these criteria. Telling the truth, Bluetooth was designed just for suiting for WPAN environment. It is proved by the fact, that IEEE accepted Bluetooth specification for the base of his 802.15 WPAN standard. This will help Bluetooth become a leader WPAN technology [1].

Besides it is suitable for WPAN devices, Bluetooth has another advantage. This technology not only replaces cables but is able to establish networks. Bluetooth supports point to point and point to multipoint connections as well. One Bluetooth network - a piconet - can be formed by up to 8 devices. These piconets can be organized to bigger network called scatternet.

3. Connection with other networks

The system was designed to be able to work together with extant network protocols just like the popular IP based TCP, UDP, WAP or object exchanging protocols (OBEX). We can use Bluetooth for voice only applications too. This co-operation with large scale of applications is realized by L2CAP (Logical Link Control and Adaptation Protocol). One of this protocol's services is protocol multiplexing allowing different protocols work over Bluetooth. Figure 1 shows the Bluetooth protocol stack [2]. We can see the several upper layer protocols over L2CAP.

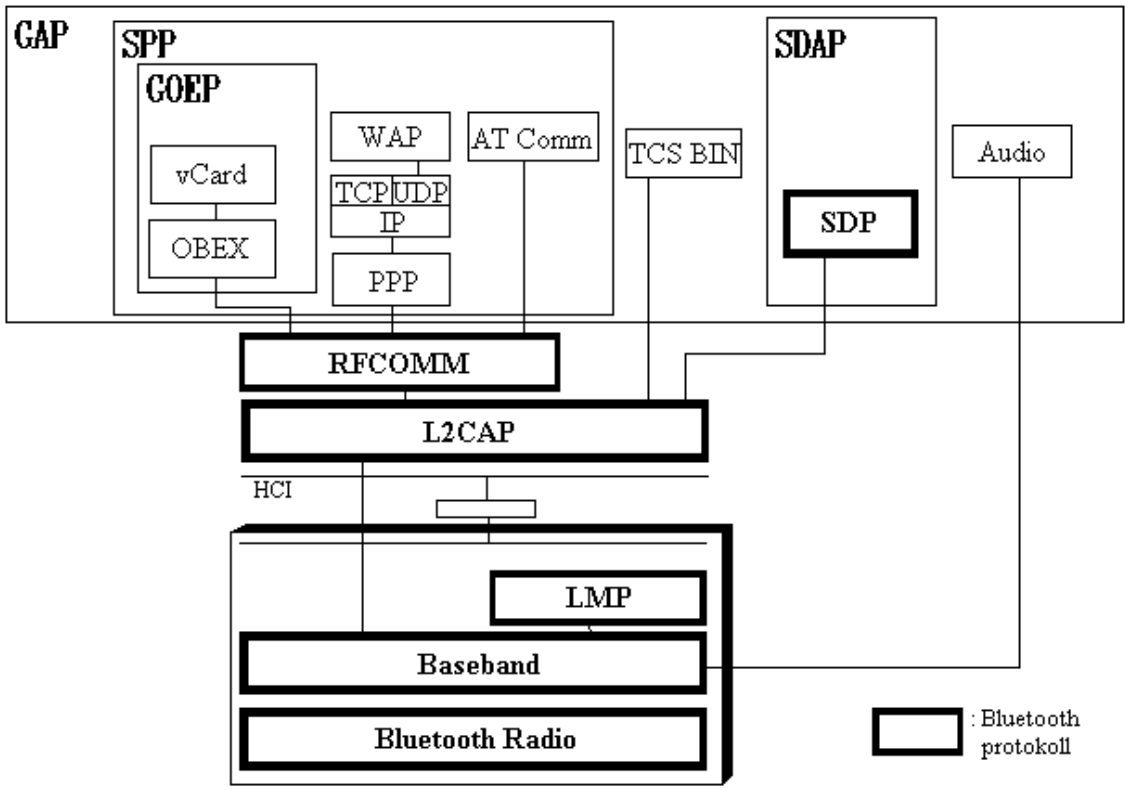


Figure 1. The Bluetooth protocol stack and profiles

Thanks L2CAP the point-to-point protocol (PPP), mobile and line Internet (WAP, IP), data exchange protocols using client-server model with different content type (OBEX, vCard, vCalendar, vNote, vMessage) can run over the Bluetooth protocol stack. We are able to control modem and fax devices also.

Another protocol called BNEP (Bluetooth Network Encapsulation Protocol) gives L2CAP the ability to handle the common network protocols, the same that are supported by Ethernet encapsulation [3]. BNEP is situated over L2CAP too. These protocols widen the broad variety of Bluetooth applications.

4. The wide scale of Bluetooth applications

We can see (Figure 2) that Bluetooth is an ideal WPAN technology to form ad-hoc networks and to access remote networks (wired or wireless) through network access points. These properties make it possible to reach a remote host connecting to a LAN or WAN easily using our personal device [4]. This way we can get information up to date, we are able to control real-time systems, the mobile device can work as a monitoring or alarming set.

Using Bluetooth for reaching another devices or networks makes it possible disabled people to have special personalized interface as a Bluetooth device to access the common computers or other equipments for their work. Their PAN devices have to be specially designed, Bluetooth can connect them to usual LANs.

To mention another application, to phone over Bluetooth means to spare the cost of the call if we want to communicate with other person having Bluetooth device placed in the area of the scatternet. We only have to pay for calls on bigger distance, which can be made e.g. over GSM.

The key protocols make the wide usability of Bluetooth real are: L2CAP and BNEP.

5. Bluetooth profiles

The Bluetooth developing group, the SIG (Bluetooth Special Interest Group) determined some basic profiles for Bluetooth. A profile is (one or more) vertical slice in the protocol stack describing the mandatory protocols and parameter ranges for different user scenarios. Using these profiles interoperability problems will be eliminated between Bluetooth devices of different manufacturers.

The used protocols are application-dependent, but the base Bluetooth protocols (Bluetooth Radio, Baseband, LMP, L2CAP, SDP) are used in every cases - excepting audio transfer,

so the implementations of the basic protocols are reusable in different use cases applying different parameters.

There are four general profiles determined covering the common user scenarios. (Figure 1.) The Generic Access Profile (GAP) handles discovery and connection establishment between unconnected devices. It is a basic profile, every Bluetooth device must support it.

The second defined profile is the Service Discovery Application Profile (SDAP). It is responsible for searching for specific or general services in the range of the Bluetooth unit. SDAP re-uses parts of the GAP.

The Serial Port Profile (SPP) emulates serial ports on two devices and connects them with Bluetooth. It is used in case of dial-up networks, fax, headset or LAN access. This profile re-uses the pattern of GAP too.

Finally the Generic Object Exchange Profile (GOEP) defines the protocols needed for applications uses object exchange. This kind of profiles can be File Transfer Profile, Object Push Profile or Synchronization Profile. GOEP uses GAP and SPP, so protocol engineers, who work out protocol stack for object exchanging Bluetooth devices, can re-use GAP and SPP implementations.

In practice for every usage model there is one or more adaptable profile.

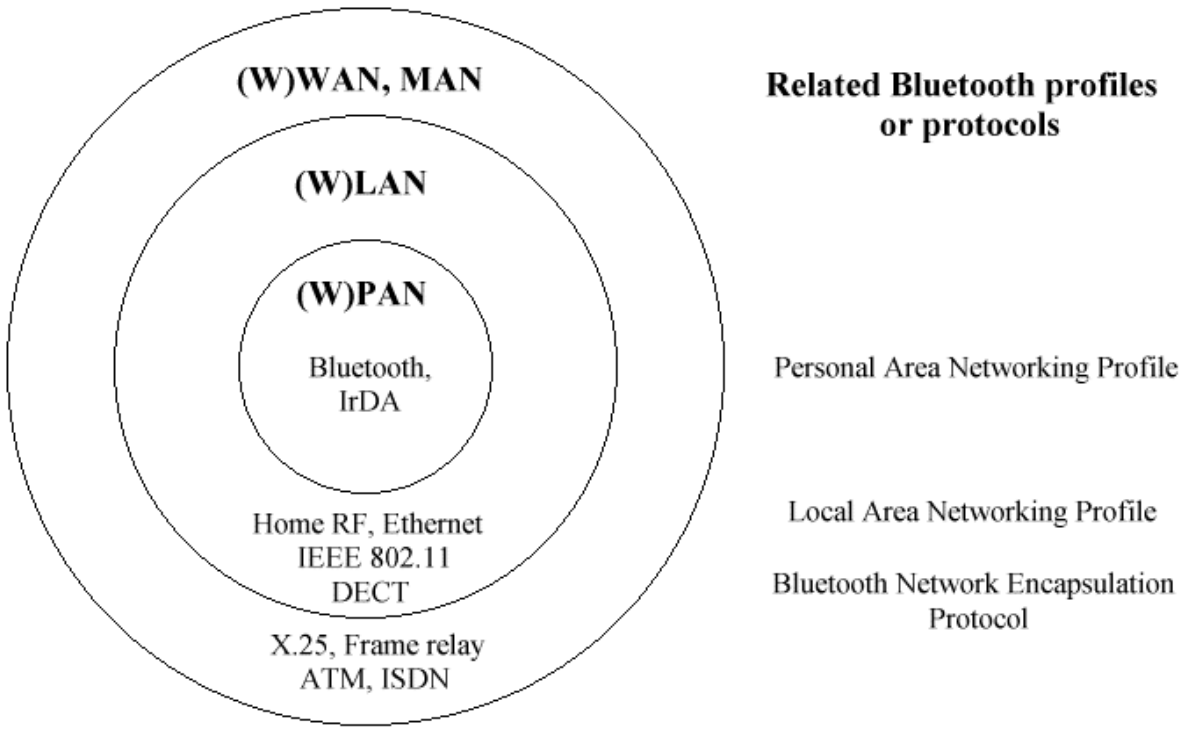


Figure 2. Bluetooth's role in interoperability between PANs, LANs and wider networks

5. Pattern based formal protocol design

There are several applications to realize them protocol engineers have to reuse these protocols with modified parameters. During the analysis reusable patterns have to be detected and designed with the help of formal languages. This kind of language is SDL, which was developed for specificating and describing distributed interactive real-time systems [5]. It is suitable for realizing protocols. The SDL description is hierarchical, it increases the transparency of the design. In the first step we have to build the model for the given protocol. It is made by means of formal patterns, in SDL this role is played by the packages. If we design the patterns, we can use them for different use cases by changing they parameters. It is very effective in case of protocols used for several different applications, like e.g. L2CAP and BNEP.

SDL is a good choice because SDL description can be converted easily into final code just like C, or we can generate TTCN description and test cases for the phase of test. With the help of this family of formal languages (SDL, MSC for sequence charts, TTCN and ASN.1 for abstract data definition) the whole life circle of protocol development can be covered.

With the hierarchy levels of SDL we can differ the static and dynamic parts of the protocols: the protocol system, the protocol entity and the protocol behavior. If we determine the reusable patterns of these levels, we can easily apply them for different use cases as we have seen it in the section dealing with Bluetooth profiles. It makes it

easy to develop the variants of protocols simply reusing the predefined pattern. This makes the developing procedure efficient especially for these kinds of protocols like L2CAP and BNEP which offer the use of wide variety of network types over Bluetooth.

References

[1] Brent A. Miller, Chatschik Bisdikian (2000). *Bluetooth Revealed: The Insider's Guide to an Open Specification for Global Wireless Communications*. Prentica Hall - PTR.

[2] Bluetooth (2001). Specification of the Bluetooth System. Core.

[3] Bluetooth (2001). Bluetooth Network Encapsulation Protocol (BNEP) Specification.

[4] Bluetooth (2001). Personal Area Networking Profile.

[5] Rolv Braek & Oystein Haugen (1993). *Engineering real time systems*. Prentice Hall Europe

An Experimental Testbed for Using WLANs in Real-Time Applications

Trygve Lunheim
Dept. of Engineering Cybernetics
NTNU, Trondheim, Norway
trygvelu@itk.ntnu.no

Amund Skavhaug
Dept. of Engineering Cybernetics
NTNU, Trondheim, Norway
amund@itk.ntnu.no

Abstract

In this paper we describe an ongoing experiment to determine the feasibility of deploying COTS wireless technology, like IEEE 802.11b networks, in specific real-time application scenarios. Wireless networks are being used in an increasing number of applications, and the focus towards consumer markets has driven the cost down. The deployment of wireless communications is desirable for certain field devices in industrial automation and process control, where wiring might not be feasible and/or not cost-effective. Ensuring real-time performance and stability is not straightforward, however. We propose to use experimental results to determine what kind of performance can typically be expected in different scenarios, thus enabling us to make some assumptions about proper deployment of these technologies.

1. Introduction

Distributed real-time systems need to deal with communications in a way that is not only reliable and stable, but also predictable within real-time constraints. In the past there was for a long time reluctance to use Ethernet in real-time applications, due to the non-deterministic bounds on packet delay in loaded networks. With the introduction of *switched* Ethernet this has changed, and today Ethernet is a popular choice in many industrial applications, because of its maturity and low cost. However, since switched ethernet inherently has a star topology in the physical layer, there may be a higher cost in cabling compared to eg fieldbus networks, which means that some of the cost-related motivation behind the use of Ethernet in field devices is lost.

Wireless network technology is an interesting choice in several areas of process control and monitoring, because of the cost or difficulty of wiring in many situations. The interconnection of mobile field devices in an industrial plant is an example of a scenario where the deployment of a wireless local area network (WLAN) would be desirable[1]. Advantages typically offered by WLANs over wired networks include mobility, flexibility in dynamic environments, and easy installation. Wireless communication networks for the consumer market have recently received much attention, and the IEEE 802.11b WLAN standard has been particularly successful in gaining popularity, thus driving the cost of equipment down.

The IEEE 802.11 medium-access control (MAC) layer defines a distributed coordination function (DCF) for best-effort asynchronous traffic, and an optional point coordination function (PCF) for supporting real-time traffic. PCF implements polling to eliminate collisions, and uses an access point (AP) for control, whereas DCF uses a protocol based on carrier sense multiple access with collision avoidance (CSMA/CA). In an implementation only DCF is mandatory, and thus the support of real-time traffic in IEEE 802.11 DCF networks has been the focus of several research efforts.

Much effort has been placed in the analysis and simulation of WLANs, and several approaches have been proposed to support mixed traffic in IEEE 802.11 by improving the MAC protocol layer ([2],[3],[4]). However, these approaches are often not available to us in the design of a distributed control system, since we have to rely on standard components provided by manufacturers and in the real-time operating system (RTOS).

To enable us to make some assumptions about the performance of wireless commercial off-the-shelf (COTS) components in a real-time application we propose to conduct a series of experiments to measure throughput and timeliness in different scenarios. By using analysis and simulations on a model of the network it is difficult to get clear results, because there are so many variables and uncertainties, eg with buffers in different protocol layers. In our opinion a good way of telling if something works, is to actually try it out.

Some other experiments conducted on IEEE 802.11b networks have mainly concentrated on best-effort throughput [5]. Another experiment on traffic over 802.11b networks is described in [6].

2. Experiment

In conducting the experiments we make use of COTS hardware that we have access to, and make use of our surroundings to represent different scenarios. Initial focus has been on IEEE 802.11b networks, but hopefully the same considerations can be used when experimenting with other WLAN technologies, like eg Hiperlan2[10].

2.1 Traffic issues

In an automated factory environment there are typically a large number of sensors, where some generate real-time data. These are sent periodically or per request to processing units over the network; this is commonly referred to as a producer-consumer problem. We will consider the real-time traffic to consist of frequent short packets, which can be transmitted across the network using eg UDP/IP.

Both one-to-one (unicast) and one-to-many (multicast) transmission between producing and consuming nodes can be considered.

2.2 Environment

The performance of any WLAN technology depends greatly on the environment in which it is being deployed. In the case of a weak signal the bandwidth will suffer from degradation. We will use the same experimental setup in different scenarios to reflect this issue.

The *shielded* environment consists of an underground room which is thus shielded from outside noise. The environment should prevent radio interference (much like a Faraday cage), and provide an "ideal" case for our experiments.

Experiments in the *rough* scenario are conducted in an experiment hall with a lot of electrical equipment, pipes and wiring, resembling a real industrial plant. This hostile environment should provide us with a certain degradation of the signal, similar to actual industrial use.

Finally the *office* environment provides us with some results in an open office environment, which is convenient for ongoing testing. This is also similar to the environment that is listed in the specifications of the network devices.

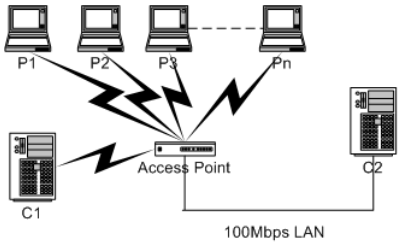


Figure 1. Experimental Setup

2.3 Experimental setup

A number of producer nodes P1, P2..Pn are set up to generate traffic meant for consumer nodes C1, C2.. By analyzing received data in the consumer nodes we can produce statistics for network throughput with varying packet size and send rate. When comparing throughput we are not so concerned with the amount of data transferred as we are with the timeliness of delivery. Packet loss because of collisions is also considered, and the wired C2 (Figure 1.) provides us with some comparison data for the wireless consumer node C1.

The 802.11b network cards used may be produced by different vendors, giving them different characteristics, and this must be accounted for. The WiFi-certification should ensure that the behaviour is somewhat similar, however.

The access points (APs) available for testing may have different features and characteristics, like eg buffer size. Using a PC to implement an AP is also possible, using Linux or similar OS with bridging capabilities. By using the PC as AP we can control the behaviour of the AP, and it is also possible to implement some prioritizing scheme in the AP for ensuring real-time QoS.

2.4 Synchronization issues

There is a need to measure the contributions in packet delay and latency introduced by different parts of the network, specifically the TCP/IP stack and the OS. One way of achieving this is to measure delays with the nodes connected to a high-speed (100Mbps) switched network, and assume that this network has negligible delay compared to the WLAN technology of interest.

By instrumenting the system properly we can then find the contribution from each part of the network, and thus get a separate reading on the performance of the WLAN.

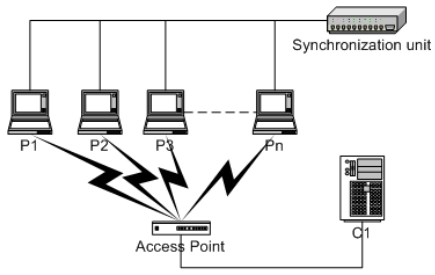


Figure 2. Synchronization of nodes

In order to flexibly be able to control the setup we will utilize a synchronization network consisting of I/O channels to each node (Figure 2.). This is necessary to achieve tightest possible synchronization of the nodes. An external unit can then signal each node when data should be generated and transmitted. By using a RTOS like QNX [7] we achieve improved control over the timing of the nodes, as well as minimizing delays caused by the OS.

2.5 Tools

The *ttcp* [8] and *NetPIPE* [9] utilities are well suited to conduct a series of tests, and can produce results to be transformed into graphical representations. NetPIPE is a particularly useful tool for visualizing network performance based on throughput and/or latencies, and it is also protocol independent, which lets us compare results with non-TCP/IP networks. A simple test using NetPIPE between two computers using 10Mbps Ethernet (a common office hub) and the same test with one of the machines (a portable computer running linux) communicating through an AP gave us a *network signature graph* as seen in Figure 3.

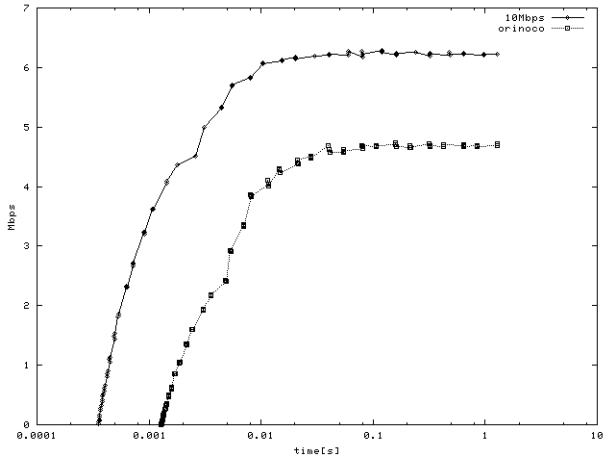


Figure 3. Network signature graph

From this graph we can easily read that common Ethernet has a lower latency (the first data point of each graph) and also a higher throughput than the 802.11b orinoco network. Analyzing blocksize vs throughput and timing are other options with this tool.

For conducting the tightly synchronized tests we need to write our own suite of utilities. This is necessary to gain full control over the time domain of the transmission.

3. Current and future work

Currently we are setting up the network for conducting the experiment, and we should hopefully see some results soon.

By using the same equipment in future experiments we will be able to compare results with other WLAN technologies, like HiperLAN2, IEEE802.11a and others.

4. References

[1] S. Cavalieri and D.Panno, "*On the Integration of Fieldbus Traffic within IEEE 802.11 Wireless LAN*", Proc. of IEEE Intl Workshop on Factory Communication Systems, 1997

[2] S. Sharma, K. Gopalan, N. Zhu, P. De, G. Peng; T. Chiueh, "*Quality of service guarantee on 802.11 networks*", Hot Interconnects 9, 2001, pp 99 -103

[3] A. Veres, A.T. Campbell, M. Barry, L. Sun, "*Supporting service differentiation in wireless packet networks using distributed control*", IEEE Journal on Selected Areas in Communications, vol. 19 iss. 10, Oct. 2001, pp 2081-2093

[4] H. Ye, G.C. Walsh, L.G. Bushnell, "*Real-Time Mixed-Traffic Wireless Networks*", IEEE trans. on Industrial Electronics, Vol. 48 iss. 5 , Oct. 2001, pp 883-890

[5] http://www.uninett.no/testnett/wlantest/ (in norwegian)

[6] T. Pagtzis, P. Kirstein, S. Hailes, "*Operational and fairness issues with connection-less traffic over IEEE802.11b*", IEEE Intl. conf. on Comm., Vol. 6 , 2001, pp1905-1913

[7] http://www.qnx.com/

[8] "*ttcp - a benchmarking tool for determining TCP and UDP performance between two systems*", http://www.netcordia.com/network-services.html

[9] Q.O. Snell, A.R. Mikler, and J.L. Gustafson, "*NetPIPE: A Network Protocol Independent Performance Evaluator*", http://www.scl.ameslab.gov/netpipe/

[10] http://www.hiperlan2.com/

Workload Balancing in Distributed Virtual Reality Environments

Michael Ditze[1], Filipe Pacheco[2], Berta Batista[2], Eduardo Tovar[2], Peter Altenbernd[1]
[1]C-LAB, 33094 Paderborn, Germany, [2]ISEP-IPP, Polytechnic Institute of Porto, Portugal
[1]Michael.Ditze, Peter.Altenbernd@c-lab.de [2]emt,ffp,bbatista@dei.isep.ipp.pt

Abstract

Virtual Reality (VR) has grown to become state-of-the-art technology in many business- and consumer oriented E-Commerce applications. One of the major design challenges of VR environments is the placement of the rendering process. The rendering process converts the abstract description of a scene as contained in an object database to an image. This process is usually done at the client side like in VRML[1] a technology that requires the client's computational power for smooth rendering.

The vision of VR is also strongly connected to the issue of Quality of Service (QoS) as the perceived realism is subject to an interactive frame rate ranging from 10 to 30 frames-per-second (fps), real-time feedback mechanisms and realistic image quality. These requirements overwhelm traditional home computers or even high sophisticated graphical workstations over their limits. Our work therefore introduces an approach for a distributed rendering architecture that gracefully balances the workload between the client and a cluster-based server. We believe that a distributed rendering approach as described in this paper has three major benefits: It reduces the clients workload, it decreases the network traffic and it allows to re-use already rendered scenes.

1. Motivation

Conceivable applications scenarios that strongly benefit from the use of VR include architectural design analysis, distributed learning environments and travel management settings where vacationers can walk through potential hotels in advance. Imagine a walkthrough scenario consisting in the simulation of a well-known commercial street to be used by residential users. Here, the expectation is to produce a high quality simulation environment that highly resembles the original. The 3D nature of the simulation should allow the users to interact with the environment in quasi real-time: change their point of view in the three-dimensional space, zoom in on details and trigger pre-recorded actions by means of hot spots. When interacting with such a virtual environment, realism depends mainly on three factors: realistic images, interactive frame rate (10 to 30 frames per second) and real-time feedback (motions, behavior, etc.).

By itself, the generation of photo-realistic images from a 3D-object database; i.e. the *rendering process* is computationally extremely expensive, and still imposes major research challenges, whereas the complexity of lighting phenomena associated to interactive usage further calls for powerful and predictable computing in order to met the user expected time constraints.

On the other hand, walkthroughs of large information spaces face the task of generating images from a model containing a huge amount of elements.

Given this a complete framework will include the integration of existing and, when required, development of new solutions to several challenging real-time problems, such as:

- real-time distributed client-server networking providing proper guarantees;
- real-time distributed computation of parallelised rendering tasks for clusters of workstations networked via commodity RT LANs (rendering engine at the server side);
- timely scheduling and execution of rendering tasks and media-players running on the client;
- timely scheduling and execution of multiple client requests (front-end server at the server side);
- the adaptation of current workload, including client-server balancing of rendering load.

2. MPEG-4 as a Client-Server connection

Traditional VR systems usually form a Client-Server architecture where the information of objects is on request transferred from the server to the client. The rendering process itself that may result in large computational overhead is then left to the client. An idea how to transform the conservative client-server methodology to form a distributed rendering approach can now be realized by using MPEG-4.

MPEG-4[2] is a common video compression standard originally targeted at video streaming applications used in environments with very restrictive bandwidth at disposal. It was developed by the Motion Pictures Experts Group and finalized as a standard in 1998. In future, MPEG-4 will be used for video streaming applications in UMTS. In contrast to preceding MPEG standards (MPEG-1 and

MPEG-2), MPEG-4 follows an object-oriented approach. Video scenes are decomposed into single arbitrarily shaped objects called audio-visual objects that are separately encoded and transmitted over the network to the client. Examples for these objects range from primitive media objects like audio or still images to complex object representation in 3D environments. Besides fast encoding mechanisms, the major benefits of MPEG-4 are its scalability in terms of gracefully adjustable video quality with regard to network capacity, reusability of video objects across different video scenes and platforms and QoS support for network service providers.

The advantages of MPEG-4 very well serve the idea of distributed rendering in VR environments. In contrast to the traditional solution where computationally expansive rendering is completely done at the client side, the server which has usually more computational power may now partly pre-compute complex rendering scenes and encode the rendered scene as a MPEG-4 audio-visual object which is then transmitted to the client. Alternatively, the rendering could also be adopted to some other client with vacant resources. The advantages of this new approach are obvious: The client is greatly relieved from the computational overhead it has to spent for rendering and the rendered MPEG-4 object qualifies for re-utilization across various clients that wish to display the same rendered scene. Moreover, network traffic is significantly reduced since a MPEG-4 object that is targeted at low bitrates will consume less bandwidth than complex object descriptions which, in traditional VR environments, still must be transmitted to the client.

3. Network Management Unit

In order for the server to determine which scenes are to be pre-rendered, a network management unit is required. The NMU keeps track of the current network workload as well as of the computational workload on all clients that participate in the VR. If the workload of a particular client exceeds the resources at disposal as determined by a Response Time Analysis (RTA)[4], the NMU may decide to re-distribute the rendering process to the server or to an alternative client with free resources at disposal. This necessitates the client to apply a RTA and a scheduling algorithm that takes the specifics of MPEG-4 and rendering into account. A scheduling algorithm and RTA for MPEG-2 streams that may be adopted to suit MPEG-4 is presented in [5].

Moreover, the NMU is also responsible for reserving the required bandwidth on the respective link in order to guarantee that MPEG-4 video objects as well as VR object descriptions can be transferred within given timing constrains. RSVP provides such a set of communication rules that allows channels or paths on the Internet to be reserved for the transmission of video and other high-bandwidth messages [6]. In order to determine weather a particular network link is still capable of transmitting additional data before a pre-determined deadline, a RTA for the network link must be performed. [7] presents such a RTA for RSVP.

4. Peer to Peer Networking

VR environments usually allow for interaction with virtual objects or other virtual persons represented through clients in the same VR environment. Apparently, in case of multiple clients that navigate the same VR environment, information that represent this environment may be redundant and respective objects, e.g. the background of a scene, may already have been rendered by other clients in the VR. Peer to Peer Networking offers a great opportunity to determine clients or servers that have already rendered this scene and stored in a MPEG-4 video object by addressing the NMU. Instead of repeating the rendering process, the client in need requests this particular MPEG-4 object through the network and displays it. Note, in each case at least the NMU knows where to find particular objects as it is responsible for the distributed workload balancing.

5. Cluster-based Rendering

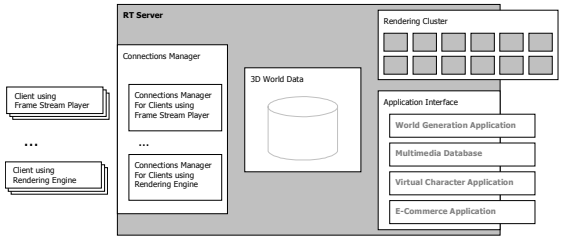


Figure 1 – A VR Server Architecture

For complex scenes or high-quality images, the rendering process is computationally intensive. This is particularly acute for a rendering server, which will have to serve multiple clients. The RT VR Server may consist of a "front-end" machine, for managing the client-server balancing and working as an interface to the NMU, and a cluster of networked personal workstations (PWS) acting as the server's rendering engine. Such a cluster will provide cost-effectiveness for both performance and scalability[8], which are main platform requirements. Additionally, maturity and robustness of Linux/RT-Linux[9] and *de facto* standardization of message-passing via Parallel Virtual Machine (PVM[10]) and Message Passing Interface (MPI[11]) are enabling the design of systems which are entirely made up of COTS technology.

However parallelism problems in rendering are usually regarded as intractable[12]. In fact, although the rendering process contains ample parallelism at different levels of the rendering pipeline, it is not easy to efficiently distribute the processing between different units, mainly due to the enormous sharing of information in the rendering process. The question of integrating parallel renderers into the broader computing environment has often been neglected, and in most cases explicitly ignored[13]. Nonetheless, diverse research works have been published focusing on parallel rendering[12][13][14].

A final important requirement is efficient real-time LAN technologies for the rendering cluster. Even though clusters of PWS are used for parallel rendering in at least one commercial rendering package[15], its actual implementation is hampered by the lack of efficient networking technologies. This will be detailed in the next section.

The Server Architecture may also supports additional client-server functionalities (including application extensions interfaces) that will not be detailed in this paper.

6. The RT Cluster Network

The RT cluster implementation must consider several issues in order to achieve the adequate behaviour/performance, namely the cluster interconnect, the message passing scheme and the operating system.

Traditionally expensive interconnects are proprietary and rely on special purpose hardware losing the cost benefit offered by the commodity market. In this class GigaNet[16] and Myrinet[17] are two of the leading interconnects for clusters of commodity computer systems.

The building block of a Myrinet network is a 16-port switching chip. It can be used to build a 16-port switch, or can be interconnected to build various topologies of varying sizes (albeit not all allow easy finding of contention-free routes). The core of the switching chip is a pipelined crossbar that supports non-blocking cut-through routing of packets. The routing algorithm is based on source-routing according to the information present at the variable-length packet header. Myrinet provides reliable, connection-less message delivery between communication end-points. This is achieved by maintaining reliable connections between each pair of hosts in the network and multiplexing the traffic between end-points over these reliable paths [18]. Simulation results showed, however, that Myrinet latency, under heavy load, suffers due to the blocking in the distributed wormhole routing scheme [19]. There is no efficient support of broadcast communication as well.

GigaNet is a connection-oriented interconnect. No message can be exchanged between communication processes until a VC has been established. Each VC corresponds to the allocation of buffer queues, routing table entries and other resources in the network and, at the host, that limits the size of the cluster. With the connection-oriented communication semantic, circuit-based switching and end-to-end flow control scheme are naturally adopted for GigaNet.

The most popular Local Area Network (LAN) technology is Ethernet. Today standards ensure bandwidths of 10-, 100- (or fast), 1000-Mb/s (or gigabit) and 10000-Mb/s and there are already discussions of 100-gigabit per second Ethernet, which could provide the next generation parallel computers with a smooth upgrade path to their communication subsystem. Ethernet, in addition to bandwidth enhancement present in the last implementations, and particularly in the full-duplex mode, allows switched access at full channel capacity without the limitation of CSMA/CD. Therefore, we believe that, given a scalable switching architecture, Ethernet can be a cost-effective solution for cluster computing.

Conventional Ethernet switches are not fully scalable because they use designs based on a backplane bus or crossbar switch, so cascading is required to build a cluster beyond the size of the upper limit imposed by the number of nodes the switch supports, and latency is increased. The spanning tree algorithm is used to calculate a loop-free tree that has only a single path for each destination, using the redundant paths as stand-by links. At present, due to the remaining lack of switch scalability we believe that applications using e.g. a conventional Gigabit Ethernet switch fabric are limited to the smaller parallel systems in which this application includes to.

The combination of message passing middleware and high-speed interconnect is one of the crucial components for building high-performance commodity clusters. But the performance of capable network technologies can be severely influenced by the overhead in the host cluster computing, as it is also demonstrated in [19]. So there have been proposals where the role of the operating system was much reduced and user applications are given direct access to the network interface, which resulted in the industry standard for user-level communication VIA[20]. Some studies quantified the impact of user-level communication against network bandwidth on the performance of a content-aware server, comparing TCP/IP and VIA over Fast Ethernet and a higher bandwidth network. Results demonstrated that reduced processor overhead, remote memory writes, and zero-copy can all provide performance gains, whereas network bandwidth is not as important [21].

7. Conclusions

This paper presented a work in progress that aims at finding a new approach for the distributed rendering in Virtual Reality environments. Unlike traditional approaches that usually apply a client-server architecture where the computationally expensive rendering process is completely done at the client side, an new idea is introduced that enables the server or some other client to adopt pre-rendering to relieve overloaded clients. It exploits the MPEG-4 standard that allows to decompose video scenes into single individual objects that are encoded and transmitted separately.

Furthermore, a Network Management Unit has been presented that determines vacant computer and network resources and distributes the workload accordingly. It exploits RSVP for bandwidth reservation and RTA as Admission Control and along with a RTA executed at each client, it guarantees QoS all along the data path form the source to sink.

A mechanism for Peer-to-Peer networking was described that allows for efficient re-use of already encoded and stored MPEG-4 objects to VR object representation.

Finally the rendering cluster issues including the internal network were discussed.

8. References

[1] ISO. *Information technology - VRML97. Information technology - Computer graphics and image processing: The Virtual Reality Modeling Language (VRML), Part 1: Functional specification and UTF-8 encoding, 1997, ISO IEC ISO/IEC 14772-1* 1997.

[2] ISO. *Information technology - coding of moving pictures and audio, Overview of the MPEG-4 standard, Final, ISO IEC JTC 1/SC29/WG11 N4668*, March 2002.

[3] Barkai, P.*Peer to Peer Computing*. Intel Press, ISBN 1-55860-475-8,2001.

[4] Burns, A., Wellings, A. *Real-Time Systems and Programming Languages (second edition)*. Addison-Wesley, 1997.

[5] Ditze, M., Altenbernd, P. "A Method for Real-Time Scheduling and Admis- sion Control of MPEG-2 Streams." *7 th Australasian Conference on Parallel and Real-Time Systems,* Sydney, 2000.

[6] Braden, R. Ed., Zhang, L., Berson, S., Herzog, S., Jamin, S.: "Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification. Request For Comments 2205" http://www.ietf.org/rfc.html.

[7] Schneider,S., Altenbernd, P. "Combining Multimedia Response-Time Analisysis and the Resource Reservation Protocol for Efficient Network Scheduling of Media Streams" *7th Australasian Conference on Paralellel and Real-Time Systems.* Sydney, 2000.

[8] Merkey, P. *Beowulf Introduction*. http://duce.mcs.kent.edu/~farrell/equip/beowulf/intro.html. 1998

[9] Epplin, J. "Linux as an Embedded Operating System" *Embedded Systems Programming. October.* RT Linux: http://www.rtlinux.org. 1997

[10] Geist, A., Beguelin, A., Dongarra, J., Jian, W., Machek, R. and Sunderam, V. *PVM: Parallel Virtual Machine*. MIT Press. 1994

[11] MPI Forum "MPI-2: Extensions to the Message-Passing Interface" *Technical Report MPI 7/18/97*, Message-Passing Interface Forum. 1997

[12] Bartz, D., Schneider, B. and Silva, C. "Rendering and Visualisation in Parallel Environments". *SIGRAPH 2000, course on Rendering and Visualisation in Parallel Environments.* 2000

[13] Crockett, T. "Beyond the Renderer: Software Architecture for Parallel Graphics and Visualisation". Institute for Computer Applications in Science and Engineering, NASA Langley Research Center, ICASE Report No. 96-75. 1997

[14] Schneider, B. "Parallel Rendering on PC Workstations" *Proceedings of 1998 International Conference on Parallel and Distributed Processing Techniques and Applications.*1998

[15] Hubbell, J. "Network rendering". *Autodesk University Sourcebook* Vol. 2, pp. 443–453. Miller Freeman. 1996

[16] Giganet, Inc. *Giganet cLAN Family of Products*, http://www.giganet.com/products, 1999.

[17] Myrinet, *Myrinet* http://www.myri.com/myrinet/overview/index.html,1999.

[18] Myrinet, Inc. *The GM Message Passing System*, http://www.myri.com, 1999.

[19] Chen H., Wyckoff P., "Simulation studies of gigabit ethernet versus myrinet using real application cores". *Presented at CANPC00 workshop of HPCA*, January 2000.

[20] *Virtual Interface Architecture Specification*, http:// www.viarch.org

[21] Carrera E. V., Rao S., Iftode L., and Bianchini R., "User-Level Communication in Cluster-Based Servers", *In Proceedings of the 8th International Symposium on High-Performance Computer Architecture*, IEEE Computer Society, Cambridge - MA, pp. 275-286, February 2002.

Common Issues in Real-Time and Media Processing

Peter Altenbernd, Ditze
C-LAB
33094 Paderborn, GERMANY
peter.altenbernd@c-lab.de, michael.ditze@c-lab.de

Abstract

In this paper, we outline the possibilities offered by real-time research results in order to handle media streaming based on the Internet Protocol (IP).

On one hand the real-time community has been producing highly advanced methods to treat time-critical processes (like priority-based scheduling). Most of this work focuses on *hard* real-time issues, so it is guaranteed that no deadline will ever be violated.

On the other hand the networking community is dominated by the recent IP technology success. Extended with corresponding *Quality-of-Service (QoS)* models with well-defined standards, IP is even used for media streaming. However, some of these models offer just very soft timing guarantees. Further, scheduling is not as advanced as in the real-time community, and end-system QoS is often neglected.

The main contribution of the paper is that we show how the results of both areas can be combined, in order to provide reliable high-quality media streaming in IP networks. We will outline to what extent available real-time theory can be used, and how it must be combined with the standards given by the networking community.

Keywords: Media Streaming, MPEG, Quality of Service, Scheduling

1 Introduction

The widespread introduction of *Internet Protocol (IP)* technology over the past few years makes it possible to offer professional and private users a host of different services. The supply of continuous data streams (video and audio (V/A)) is an increasingly important part of this. However, these media involve an enormous volume of data and jitter-free display involves time restraints.

Typical examples of time-sensitive applications of this kind are: e-commerce applications (like virtual shopping malls), video-on-demand (VoD), teleconferences, IP telephony, distributed architecture, story board design in film production, TV broadcasting, training courses (CBT). Providing QoS guarantees to these scenarios, increases user acceptance in a drastical way.

A signal with six digital TV channels (typical for satellite technology) requires a transmission capacity (with MPEG-2 compression) of 19.2 Mbit/s, which means that it takes about 3 Mbit/s to transmit a movie of PAL quality. Even if data volumes on this scale for the IP domain appear largely wishful thinking for the moment, they will certainly be achievable in the foreseeable future as network technology brings progressive improvements, and they are therefore not part of the problem area dealt with here.

Whether or not to include time response in this context presents a quite different yet fundamental problem. Traditionally, IP technology only includes strategies that cannot provide any delivery guarantees whatsoever for the media streams transmitted (Quality-of-Service (QoS)). With today's pure IP technology, packet losses and time scatter often lead to disturbances with the transmission of V/A material in the form of losses and jitter. So, the least thing to do is to add a QoS concept.

This work offers a new concept for the transmission of such time-critical data with *Quality-of-Service (QoS)* constraints via IP technology. Special focus is on the efficient delivery of MPEG-2 streams encoded with variable bitrates including both the network and the end systems. The novel approach presented here takes advantage of classic real-time experience which considerably increases resource exploitation compared to classic transmission methods. The resulting benefit (i.e. applications gain either the same performance with less cost, or performance increases on the same cost level) can be exploited by a number of commercial applications.

In the following paper we describe how packet losses and time disturbances can be prevented by reserving bandwidth in dedicated subnetworks (using known QoS models). Particular attention is devoted in this connection to the use of techniques from classic real-time theory. Our techniques will allow resource capacities to be utilized considerably more effectively. While doing this we actually employ hard real-time methods for handling a soft real-time problem.

The rest of the paper is organised as follows. In the next section, we give a brief introduction to the application context and existing QoS models. In Section 3 we show how advanced real-time theory can be combined with existing QoS models, in order to provide reliable high-quality media streaming in IP networks. We will outline to what extent available real-time theory can be used, and how it must be combined with the standards given by the networking community. In Section 4 we give our conclusions.

2 QoS Models and Media Streaming

Two different approaches have been taken in the past in order to be able to provide QoS guarantees in IP networks. Both have been defined by the Internet Engineering Task Force (IETF): *Integrated Services (IntServ)* [3], *Differentiated Services (DiffServ)* [2]. Both approaches look very promising as regards their potential use in practice.

IntServ [3] is based on the principle of reserving bandwidth for each data stream in the system. With an exclusive bandwidth it is possible to provide valid QoS guarantees, since mutual disturbances are ruled out. To achieve this, all network elements (including transmitters (servers), various routers, receivers (clients)) that are involved in the transmission process must support the mechanisms needed for this. Before a data stream is transmitted, it is always preceded by an admission control that checks whether all elements still have sufficient capacity. Transmission is only allowed if this applies everywhere. The advantage of guaranteed QoS with the IntServ approach is offset by the disadvantage of poor scalability, since it would appear to be impossible to administer the status information for several thousand streams (as is possible in the Internet) on every router. IntServ is thus more suitable for intranets or extranets.

The **DiffServ** [2] approach counters the problem of scaling by combining data streams into particular priority classes. Admission control is handled in a heuristic fashion by a central entity (called Bandwidth Broker), as opposed to the accurate path-oriented view of IntServ. Since detailed status information is not transmitted for any of the streams, it is unfortunately not possible to provide any real guarantees with this approach, although transmission quality and efficiency are improved significantly.

Today, the DiffServ approach is the focal point of general research interest for the network community. Only a few players are going for differentiated handling of global (Internet) and local area networks of limited size (intranet/extranet). The work described here is, however, concentrated particularly on these local area networks using IntServ, because they are subject to their own administration and have become increasingly important in recent times. Examples of this are networks in the field of e-commerce, edutainment and networks of companies in multimedia or film production. Networks of this type do not suffer at all from the problem of scaling and can be handled very efficiently, as our work demonstrates.

A suitable signalling protocol, the *Resource Reservation Protocol (RSVP)*, has been developed (likewise by the IETF) as a means of reserving resources for data streams as part of the IntServ concept. RSVP is not responsible for the actual implementation and utilization of the reserved bandwidth. These functions (e.g. control of the time sequence (scheduling) of individual data packets) are performed by corresponding network modules that can be designed relatively flexibly. Unfortunately, RSVP is bandwidth-oriented, which can result in very inefficient utilization of the capacity of resources, as is described in the following.

V/A streams are coded using compression techniques such as *MPEG-2* (from the Motion Pictures Expert Group) in order to enable them to be transmitted digitally. These techniques usually achieve their highest compression rate in conjunction with a *variable bit rate (VBR)*. However, this results in considerable size differences between the largest image and the other images in a video stream. In the examples we looked at, we determined that the average size of images is under 35% of the maximum [1]. Unfortunately, it is this maximum sized image that has to be taken as the basis for the RSVP reservation (peak-rate allocation). In our example, this means that 65% of the reserved resources cannot be utilized.

3 Addressing Media Streaming with Hard Real-Time Methods

We solve the above mentioned efficiency problem by using scheduling and analysis techniques from the domain of classic real-time systems.

In the past, real-time systems were considered predominantly for control applications such as anti-skid systems in vehicles. However, it is now widely accepted that the supply of a video stream with 25 images a second presents fundamentally the same problem as the implementation of a periodic control algorithm that continually calculates new manipulated variables (e.g. braking pressure with anti-skiding) at fixed time intervals. Naturally multimedia systems are not comparable with systems where safety is a critical factor, but service providers are nevertheless under strong pressure to guarantee promised QoS in practice. Generally speaking, it can be said that a good real-time design does not only offer guarantees a priori; it also frees the developer from the need to compensate for time problems by using overdimensioned hardware. The idea presented here could therefore potentially bring about a sustained improvement in the efficiency of IP components.

Novel mechanisms for admission control and packet scheduling on routers and servers hold the key to boosting efficiency, since a bandwidth-oriented procedure does not allow access to over-reservations which are lost for other data streams. The new real-time techniques use detailed information (i.e. other parameters) about the V/A streams. If time constraints permit, data packets are therefore buffered.

There are a couple of individual problems which arise with the realization of the new concept, as discussed in the following sections.

3.1 Parameters in the RSVP Scenario

Real-time theory like Response-Time Analysis, which we use for admission control (see Section 3.2), is based on the knowledge of a set of parameters, such as *period lengths, transmission times, deadlines*, etc.. These parameters completely differ from those used the network community to describe traffic flows.

We therefore elaborated methods that derive the above mentioned parameters, which is fairly different from extracting them from a control programs (see Section 3.3). Furthermore, it was not planned for RSVP to support these parameters, so we designed on a scenario in which RSVP will be used (but not expanded) to transport them [6].

3.2 Scheduling and Admission Control

Scheduling of network traffic is not a direct function of the RSVP. Instead, one possibility is to modify appropriate network modules, such as the widely used Class Based Queuing (CBQ) [5] and server packetizers, in which a typical, priority-based real-time scheduler can be implemented. A scheduler assigns the priorities of individual packets according to their urgency. There are already a large number of different heuristic approaches to assigning priority (e.g. Earliest Deadline First (EDF)), all having the object of providing optimum treatment as far as possible.

New streams that enter the system have to undergo admission control, in order to make sure that the available resource are not overbooked. In its ordinary form (i.e. the approach taken in the RSVP scenario), admission control merely adds up bandwidths. If the total exceeds the available capacity, no further bandwidth can be assigned. According to the properties of the stream (like VBR) the amount of bandwidth needed is determined by using the *Token Bucket Model.*

By contrast, our procedures are based on *Response Time Analysis (RTA)* [6]. In our concept, admission control has to analyse the time response of all streams in the system (while simultaneously taking account of the scheduling method used). This is fully compliant to the RSVP scenario presented in Section 3.1, and we could show that this operation model is more efficient than using the Token Bucket Model. Further, our way of dealing with scheduling deals with both the network and end systems [4].

3.3 Worst-Case Execution Time Analysis of Media Streaming

Scheduling and admission control both require input parameters that need to be specially calculated. To derive these values for the network is relatively easy. However, in view of the particular difficulty of predicting calculation (i.e. for encoding and decoding) and transmission times, we apply techniques used in the analysis of *worst-case execution times (WCET)* [1].

Ordinary WCET tools estimate the worst-case execution time of a given arbitrary control program. In contrast algorithm for decoding or encoding are known in advance, so our tools try to estimate WCET values from the knowledge of the streaming data.

In turn, WCET analysis results and concepts can be used for both end-system HW dimensioning (e.g. amount of memory) and SW configuration (e.g. possible frame-rate). This is particularly useful, for answering questions like *"To what extent is my PC capable of doing SW MPEG encoding?"*.

3.4 End-System QoS

Even though there is a way to allocate sufficient bandwidth on network links, timing on end-systems

has not been considered yet, like in the case of a VoD Server handling more than just one stream at the same time. Consider an example consisting of two streams: *Stream A* (10mbps) and *Stream B* (20mbps) for which network bandwidth reservation has been made. Given that their delivery imposes a load of close to one-hundred percent on the server, this means that a non-weighted CPU/disk scheduling would offer just 15mbps for each stream, so *Stream B* cannot be given the guarantee it needs. Hence, a concept for mapping reservation requests to the end system's task model [4] is needed, which requires employment of an real-time operating systems.

This problem is very similar at streaming clients, even if there is normally just one stream. However, there are other system tasks competing with the access to resources. This even holds of unconnected devices like digital VCRs and settop boxes.

4 Conclusions

The work outlined in this paper describes the use of classic real-time techniques in the field of video/audio (V/A) delivery under Quality-of-Service (QoS) constraints in Internet Protocol (IP) networks. We showed that it is possible to apply these techniques all along the path from the sender to the recipient of a V/A stream, while being conform to commonly used networking standards. The focus is on intra- and extranets, which are becoming more and more important, offering both higher efficiency and a higher degree of predictability than ordinary approaches.

Our future work will also address the use of IP in the context of hard real-time control. For doing that, available QoS models could be used. However, those models were designed for media processing data, which differs a lot from control data.

References

[1] P. Altenbernd, L. Burchard, and F. Stappert. *Worst-Case Execution Times Analysis of MPEG-2 Decoding.* In *12th Euromicro Conference on Real Time Systems*, 2000.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. *An Architecture for Differentiated Services.* Technical Report RFC 2475, December 1998.

[3] R. Braden, R. Clark, and S. Shenker. *Integrated Services in the Internet Architecture: an Overview.* Technical Report 1633, 1994.

[4] M. Ditze and P. Altenbernd. *A Method for Real-Time Scheduling and Admission Control of MPEG-2 Streams.* In *The Seventh Australasian Conference on Parallel and Real-Time Systems (PART2000)*, 2000.

[5] S. Floyd. *Notes on CBQ and Guaranteed Service*, 1995.

[6] S. Schneider and P. Altenbernd. *Combining MRTA and RSVP for Efficient Network Scheduling.* In *The Seventh Australasian Conference on Parallel and Real-Time Systems (PART2000)*, 2000.

Distributed Video on Demand Services on Peer to Peer Basis

Chris Loeser Peter Altenbernd Michael Ditze Wolfgang Mueller

C-LAB
Fuerstenallee 11
Paderborn, Germany
Loeser@c-lab.de

Abstract

Within this paper we propose architecture ideas on a distributed Video on Demand network basing on peer to peer technology. I.e. each peer offers video streams to other peers and may receive a video stream from another peer simultaneously. This results in an optimization problem depending on different factors which we approximate with the help of an simulation environment. Our approach bases on a peer to peer framework by Sun called Project JXTA which provides a set of protocols.

1 Introduction

The base idea of this proposal is to build up a local video-on-demand service without the need of a central storage-server. Such a scenario could occur in hotels. On one hand a central video server stores a couple of movies and offers them to set-top boxes distributed in the hotel rooms. There are two obvious disadvantages: For just a few parallel video-streams the server needs significant performance which results in high costs building up the whole infrastructure. Furthermore scalability seems to be a problem. On the other hand an alternative is the usage of peer-to-peer techniques: Each set-top box in the rooms gets a (relatively) small harddisk and each peer offers and requests movies at the same time. The movies are individual and no multicasting streams.

Up to now our work is in an experimental status. So far we have built up a video streaming testbed shown in Fig. 2. Furthermore the peer to peer software models are in a initial status (Fig. 3). At this point of time we mostly deal with the *P2P data placement simulation environment.*

The outline of this proposal is as follows: Within the next Section we describe general P2P techniques in the Internet followed by the P2P middleware *Project JXTA* by Sun Mic. Section 3 describes the general architecture of the P2P video storage network and finally gives a short overview of the data placement simulation environment.

2 Peer to Peer Networks

As the web continues to expand its scope to wireless devices and sensors, its growth is expected to explode to billions of new devices[2]. The popularity of the web has also demonstrated its limitations. Denial of service attacks have shown its fragility and lack of resilience to simple attacks. Services like DNS have created centralized dependencies and constrained the Internet's growth. The computing model on the web is primarily based on a client/server model where information and services are published at well-known and fixed locations (URLs). Such addressing, along with centralized web sites, have created single points of failure and bandwidth bottlenecks to popular sites: Hot spots become hotter while cold locations remain cold. A more decentralized and self-adapting computing model has been proposed by systems such as *Freenet*, *Gnutella*, and *FreeHaven* for addressing many of these limitations and have taken advantage of the increasing bandwidth, processing, and storage available on devices connected at the edge of the Internet.

The main concept of peer-to-peer computing is that each peer is client and server at the same time. Each peer may release and allocate

- **processing power**: like at *Seti at home* a bunch of peers performs a distributed computation.
- **data storage**: data is not owned by a particular member or server, but is passed around, flowing freely towards the end subscribers. When member demands for some data increase, more copies will be propagated and replicated within the community. Otherwise, fewer copies will be available as existing copies slowly disappear and are replaced by more popular data.
- **control**: each peer can offer the possibility of being controlled or illustrate monitored data. This is especially dedicated to sensors or small embedded devices.

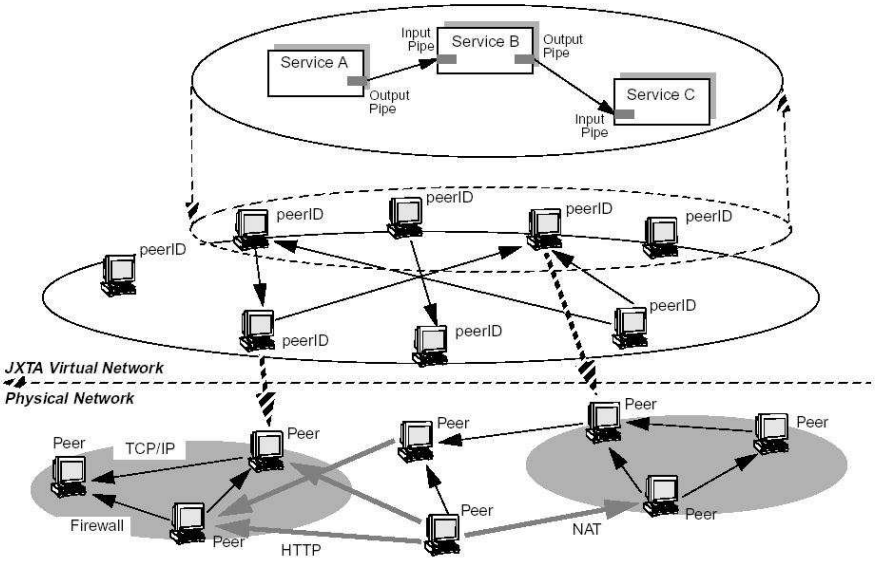


Figure 1: Jxta Virtual Network [10]

Peers have direct connection to other peers avoiding communication via mediating servers. The peer (or peer group) community as a whole is supposed to ensure the protection and persistency of data through its unique ability to adapt, resist, and protect data by scattering multiple copies within the community boundary. Community members tend to interact more heavily with their neighbors for searching and accessing information, reducing overall network traffic and data latency, and leading to a better utilization of available bandwidth. Chaining and delegation capabilities enable members to forward requests to their neighbors for searching data beyond their own view. Within a community, each member can access the entire community's knowledge. There is no single, centralized search engine; every member contributes to a search.

The described P2P architectures don't only make sense within the scope of the internet. They can easily be adapted on LANs/Intranets: also here it is reasonable to distribute content to avoid the need of centralized servers creating hot-spots.

Project JXTA

Project JXTA is an open-source project originally conceived by Sun Microsystems [8] and designed with the participation of a small but growing number of experts from academic institutions and industry. JXTA was initiated to standardize a common set of protocols for building P2P applications. Prior to this time, existing peer-to-peer systems were built in isolation, delivering a single type of service, and employing protocols incompatible with other services. For example, *Gnutella* defines a generic file sharing protocol and *Jabber* defines an instant messaging protocol, but none of these protocols are interoperable. Each system creates its own P2P community, duplicating efforts in creating software and system primitives required by P2P systems, such as managing the underlying physical network (e.g. dealing with firewalls, peer discovery, and message routing). Project JXTA attempts to define a generic network layer usable by a wide variety of P2P applications. It is designed to be independent of programming languages (e.g. C, C++ or Java), system platforms (such as the MS Windows, UNIX, Linux etc), service definitions (such as RMI and WSDL), and network protocols (such as TCP/IP or Bluetooth). The JXTA protocols have been designed to be implementable on any device with a network heartbeat, including sensors, consumer electronics, PDAs, appliances, network routers, desktop computers, data-center servers, and storage systems.

The Project JXTA protocols create a virtual network on top the existing physical network infrastructure of which services and applications are built (cf. Fig. 1). The developers designed this virtual network layer to be thin and simple, but with interesting and powerful primitives for use by services and applications. The main purpose of JXTA virtual network is to hide all of the complexity of the underlying physical network topology, and provide a uniform addressable network for all peers in the network. The JXTA virtual network allows a peer to exchange messages with any other peers independently of its network location (firewalls, NATs or non-IP networks).

It standardizes the manner in which peers discover each other, self-organize into peergroups, advertise and dis-

cover network resources, communicate with each other and monitor each other. The network transport layer is built of a uniform peer addressing scheme based on peer ID, relay peers that relay messages between peers, and a binary message format to transport binary and XML payloads.

3 Video-P2P-Network

Consider distributed video peers for storing and streaming videos where single video processing machines are connected to a network of distributed peers connected via switches and routers(cf. Fig. 2). This peer interconnection provides a virtual video server to the player application.

The management for the virtual server can be implemented by a JXTA-based middleware which ensures on the one hand *Quality of Service* for video streaming and manages on the other hand the storage and distribution of recorded videos. Each peer offers video content and may play videos from other peers by streaming simultaneously.

Details of streaming, rerouting and reallocation of storage may be completely managed and thus hidden by the middleware. The user only deals with one virtual video server and does not have to consider any network details.

In Fig. 2 you can see the raw architecture of our testbed. For our experiment we have connected peers to 3 LANs which is managed by a fast ethernet switch each. The LANs are connected to routers. Due to the experimental status the routers in the network are Linux machines. For our experiments, each peer locally stores a small number (up to now 4-7 videos) and the disk quota of each peer is limited to just a few GByte. Movies are additionally saved redundantly in order to have most possible fast access. How often movies are saved redundantly is presently evaluated by the data placement simulation.

Each peer holds a JXTA instance but also on the Linux-Router C a JXTA peer is running. Hoever here is also a so called *Rendezvous Service* working. Due to the fact that there are several subnets it is not possible for the peers to get aware of each other. Because of this the Rendezvous Server stores references of the individual peers and the peer content. However the data exchange just takes place between the individual peers. Though we need this kind of server this model does not stand in contrast to P2P models. Due to [2] these models can and do coexist.

Due to the fact that the number of peers in the network is not too large we decided to select IntServ supported Quality of Service [1]. The QoS support is achieved by the use of *traffic control (tc)* [7] [6]: The network interface is assigned to a *Queuing Discipline.* The traffic going out does not line up in a standard-FIFO rather in a *Weighted Fair Queue (WFQ).*

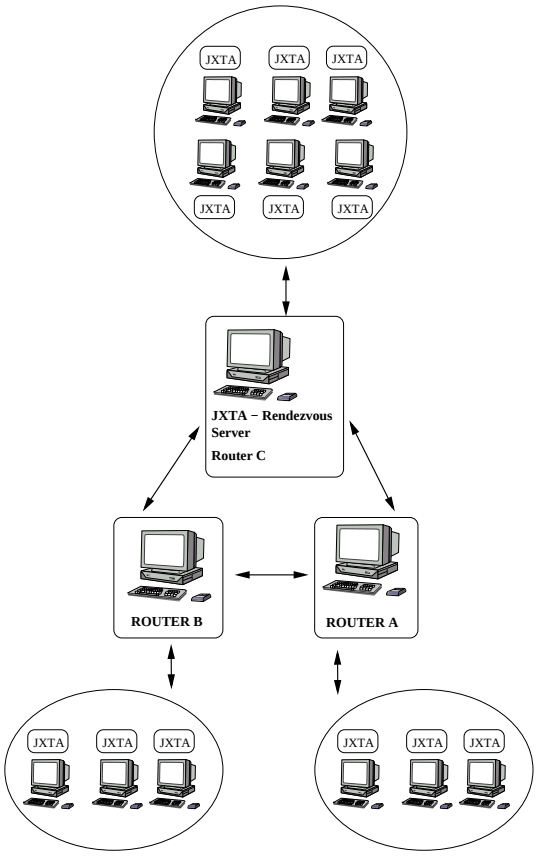


Figure 2: Architecture of the P2P-VoD testbed

3.1 Short Protocol Overview

In Fig. 3 you can see the protocol stack which shall be implemented on every peer. The *Player Application* directly interacts with the P2P instance in order to control the video stream. The P2P instance consists of several protocols which mostly are parts of the JXTA framework. Their detailed tasks are explained in [9] and [5]. Just the *Video Allocation Protocol* and the *Peer Discovery Protocol* shall be mentioned here.

- The Discovery Protocol (as part of JXTA framework) is responsible for the peer to get aware of the distributed content of the other peers. This presumes that each peer lets the rendezvous server know about its local content. This is realized by publishing *Advertisements.*

- When a user intents to view a movie the *Video Allocation Protocol* locates the most efficient source with the information delivered by the *Peer Discovery Protocol.* The RSVP protocol then reserves bandwidth along the route. We are presently implementing the VAP.

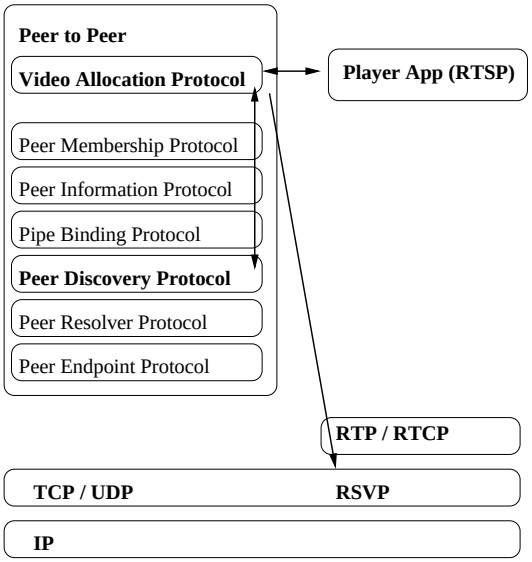


Figure 3: Protocol Overview

3.2 Simulation Environment for P2P Data Placement Strategies

In Fig. 4 you can see our Simulation Environment to evaluate different video placement strategies.
This optimization problem depends on demand distribution of the individual users, bandwidth, probability of peer failure and wether often requested movies are in the local subnetwork. The simulation applies the OSGI layer model. Besides different IP routing protocols (OSPF, RIP, DPS [3]) it is possible to reserve bandwidth resources simulating RSVP.
Later on it could make sense (when causing lower resource usage)if a serving video peer is changed at runtime to another one which is holding the redundant data.

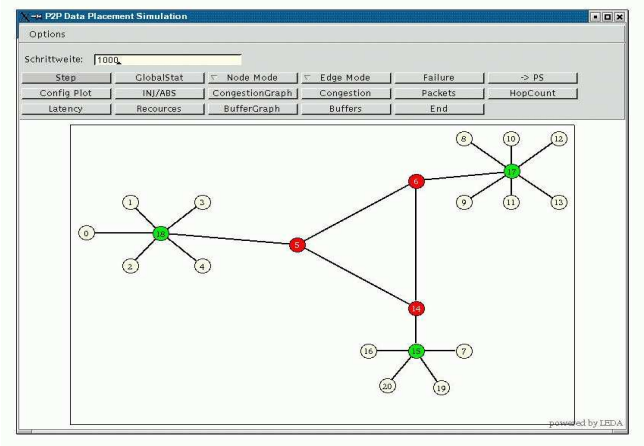


Figure 4: P2P Data Placement Simulation Environment

4 Conclusions & Future Work

Within this proposal we presented an architecture of an P2P-VoD-Network. Due to the fact that our work is still in progress we expect further results concerning the simulations within the next several month. The so gained strategies are then integrated in the Video Allocation Protocol. Moreover it might be useful to partition the video content.

Acknowledgements

Parts of the work described herein is funded by German Ministry of Technology and Research [4].

References

[1] G. Armitage. *Quality of Service in IP Networks*. Macmillan Technical Publishing, 2000.
[2] D. Barkai. *Peer-to-Peer Computing*. Intel Corporation, 2001.
[3] P. Berenbrink, A. Brinkmann, and C. Scheideler. Distributed path selection for storage networks. In *Proceedings of the 2000 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'2000), Las Vegas, USA*, pages 1097–1105, June 2000.
[4] Information Technology for European Advancement (ITEA). *Middleware for Virtual Home Environments*. `http://www.vhe-middleware.org`.
[5] S. Li. *Jxta. Peer-to-Peer Computing with Java*. 2002.
[6] G. Maxwell, R. van Mook, M. van Oosterhout, P. Schroeder, and J. Spaans. *Linux Advanced Routing and Traffic Control HOWTO*, 2001. `http://www.tldp.org/HOWTO/Adv-Routing-HOWTO.html`.
[7] S. Radhakrishnan. *Linux - Advanced Networking Overview*. University of Kansas, 2000. `http://qos.ittc.ku.edu/howto/`.
[8] Sun Microsystems. *Project JXTA*. `http://www.jxta.org`.
[9] Sun Microsystems. *JXTA Protocol Specification*, 2002. `http://spec.jxta.org/v1.0/docbook/JXTAProtocols.html`.
[10] B. Traversat, M. Abdelaziz, M. Duigou, J.-C. Hugley, E. Pouyoul, and B. Yeager. *Project JXTA Virtual Network*. Sun Microsystems, Februrary 2002.

Dynamic Real-Time Bandwidth Sharing Algorithm for Broadband Multimedia Communication Systems

Yacine Atif
Department of Computer Science
United Arab Emirates University
Al-Ain 17551, U.A.E
Yacine.Atif@uaeu.ac.ae

Abstract

Distributed multimedia information systems take the form of client-server architecture where the server acts as a repository for multimedia documents, which are compound documents composed of synchronized media objects. When transmitted across the network by the server, such documents require Quality of Service (QoS) guarantees to ensure a certain level of Quality of Presentation (QoP) at the client side. While broadband networks such as ATM provides end-to-end QoS guarantees, through dedicated network channels, a judicious transmission scheduling algorithm is required to decide which channel should transmit which media object and when, to achieve the required level of QoP. In this paper, an efficient real-time scheduling algorithm for delivering multimedia documents on broadband networks is discussed. This server-based scheduling algorithm dynamically adjusts itself to environmental parameters such as the available network resources and the size and the quantity of the requested multimedia documents. The technique we propose automatically controls and allocates the algorithm run-time cost, in order to minimise the transmission deadlines violation of media objects at due to the algorithm's overhead.

1. Introduction

Emerging technologies in fibre-optic and digital broadband network, such as Asynchronous Transfer Mode (ATM) and gigabit Ethernet have led to the evolution of distributed multimedia information systems (DMIS). DMIS allows applications to transmit heterogeneous traffic types such as video, audio, images and text. In such systems, a multimedia database server acts as a repository of multimedia documents, which are composite documents, composed of heterogeneous multimedia objects. These objects are temporally linked to each other according to some synchronization requirements as shown in Figure 1.

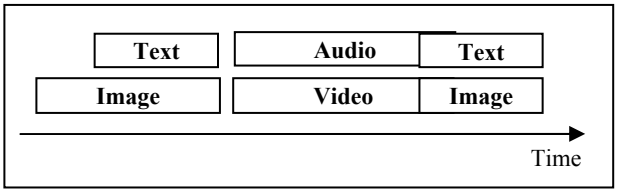


Figure 1. Multimedia document composed of synchronized multimedia data

Thus multimedia documents are pre-orchestrated documents, which are accessed by clients to be viewed locally. Note that clients do not wish to download the whole document to their local disk, as clients' computers may not have the storage capacity of the server. Clients may however buffer the current segment of the multimedia document being played.

The playback of multimedia documents at the client side should maintain the synchronization requirements within that document for a smooth display of its content. This constraint will only be satisfied if multimedia data that compose the multimedia document reach the client side at or before their scheduled playback time. Such constraints raise several problems, requiring a real-time operating system support and Asynchronous Transmission Mode (ATM) networks which distinctive QoS feature is

the provision of traffic contract through appropriate bandwidth to different types of traffic.

In this paper, we propose a dynamic scheduling algorithm which runs at the server side with the objective of assigning real-time SIUs to broadband network channels. The proposed algorithm plans the scheduled times of SIUs' transmission and the transmitting channels. Once a schedule has been built, it is delivered to the ATM subsystem layers for transmission. The delivered schedule is correct in the sense that no SIU will miss its playout deadline when it reaches the client side. The remaining sections of the paper are organized as follows. Section 2 states the problem and set the objectives of the research work discussed in this paper. Section 3 reveals our scheduling algorithm. Section 4 provides a performance evaluation of the algorithm and finally Section 5 concludes the paper with a summary of results and a proposed future work.

2. Problems and objectives

The objective of the proposed scheduling algorithm in this paper is to preserve the Quality of Presentation (QoP) requirements for quality playback of a multimedia document using the QoS guarantees of a broadband network. The proposed algorithm in this paper strives to map the user-requested QoP to the available network QoS. If the available QoS cannot accommodate the requested QoP, connection is refused. Similarly, if QoP requirements cannot be met, the document playback is interrupted. QoP is expressed in terms of threshold parameters which value need to be satisfied in order for a playback of the corresponding multimedia document to take place. For continuous media for instance, depending on the required quality, a user can quantify the acceptable data loss in a network environment. A fraction of the media objects can be dropped without degrading the required playback quality. This fraction consists in the ratio of the required rate of presentation to the nominal one. For example, if the user can tolerate a presentation rate of 20 frames per second for a video object (instead of 30 frames per second for NTSC quality video), every third frame can be dropped. A set of QoP parameters to specify the desired quality of presentation of multimedia information were defined as shown below. Table 1. Table 1 QoP parameters

This means that, in an audio clip, 1 audio SIU can be dropped for every 50 audio SIUs, and in a video clip, 1 video SIU can be dropped for every 10 video SIUs. The above QoP parameters may vary from application to application especially with respect to the content of the video SIUs. To guarantee their satisfaction, QoP parameters requirements need to be mapped to the network QoS parameters.

Each SIU_i to be transmitted on channel C_j is characterised by the size of the SIU s_i and deadline d_i. We define an SIU-to-Channel assignment (SIU_i C_j) to be feasible, if SIU_i transmitted on channel C_j reaches its destination before its playout deadline. An SIU that is feasible on one channel may not be feasible on another channel, even if this SIU is the only one assigned to the channel for transmission. This may be due to the channels' bandwidth or the size of the SIU. If a schedule includes all the SIUs, then it is considered to be complete. Otherwise, it is referred to as a partial schedule.

3. Dynamic real-time scheduling

Scheduling can be represented as the problem of incrementally searching for a feasible schedule in the graph $G(V,E)$, that represents the solution space [2]. G in our representation of the problem is in the form of a tree as shown in Figure 2. The vertices $v_i \in V$ in G represent SIU-to-channel assignments (SIU_i,C_j). A partial path (v_i, v_j, …,v_k) from the root v_i to a vertex v_k in G represents a partial schedule. The partial schedule includes all the SIUs that have been scheduled so far and is represented by v_i, v_j, …,v_k. The edges represent extending the partial schedule by one more SIU-to-channel assignment. To extend a schedule (or path), one SIU is assigned to a channel at a time, and the feasible schedule is incrementally built. Thus, schedule construction proceeds from one level in G to the next level. Complete schedules, if they exist, will be at the leaves of G. The incremental schedule construction allows a dynamic algorithm to produce a partial schedule that is feasible at any point during scheduling process. To choose a new SIU to add to the schedule and to assign that SIU to one of the channels, we need to evaluate and compare a set of candidate vertices with one another in G. The candidate vertices are stored in a candidate list, CL of all feasible SIU-to-Channel assignments for the SIUs considered so far. Thus, feasibility checks are applied to different vertices in G to identify the valid SIU assignments that can be added to the partial schedule. The search makes use of a heuristic function, which prioritizes the nodes to be explored based on SIUs' Earliest Deadlines at the SIU dimension and channels' Earliest Available Time at the channel dimension.

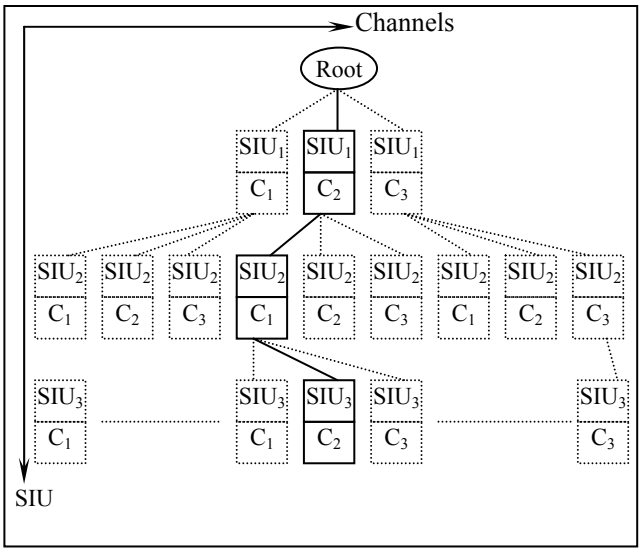


Figure 2 Search-based representation of scheduling

In this algorithm, SIUs arrive continuously and form the input of the scheduler, which selects the best assignments SIU-to-channel on-line, and distributes the SIUs to their corresponding channels in the system for transmission. Hence, the scheduler switches between two states: a scheduling phase followed by SIUs' transmission phase. The scheduler runs in parallel with the transmission of SIUs by the network channels. Thus, while the network channels are transmitting the batch of SIUs from the previous phase, newly arrived SIUs are scheduled and later they are added to the transmission queues of the assigned channels as shown in Figure 2.

The time at which the problem results are produced is critical in any real-time systems. The time consumed to produce those results is equally important especially in dynamic environments [2]. The more time spent in building a schedule, the better will be the quality of the resulting schedule. However, the delay in delivering the schedule may result in missing the deadlines of scheduled SIUs. To resolve this dilemma, we propose in this section a time-limit generation formula based on which the scheduling overhead is upper-bounded to avoid missing SIUs' deadlines because of scheduling time. As such, we have factored in a scheduling quantum $Q_s(j)$ (i.e. time-limit) in the scheduling algorithm that changes dynamically in reaction to various parameters such as slack time, SIUs arrival rate and actual channels' load, which affect the value of the time to allocate to a scheduling phase. For instance, a large slack-time, a low arrival-rate and a high channels' load suggest a large quantum value. When the channels load is high, even if the scheduler is able to produce a feasible schedule quickly, the channels are not able to transmit the assigned SIUs immediately. On the other hand, a short slack-time, a high arrival-rate and a low channel load suggests a small value for the time quantum. By bounding the scheduling time, the algorithm ensures that SIUs do not miss their deadlines because of scheduling overhead. The criterion to control the allocation of time quantum $Q_s(j)$ of a scheduling phase j is shown in Figure 3.

$$Q_s(j) <= \min(\max(min\_slack, min\_load), k/\lambda)$$

where: $min\_slack = \min(Slack_i \mid SIU_i$ element of $batch(j))$

$min\_load = \min(Load_k \mid C_k$ element of channels))

Figure 3.. Criterion for allocation of scheduling-time.

The expression in Figure 3 considers the maximum of *min_slack* and *min_load*. By taking *min_slack* into consideration, we ensure that no SIU will miss its deadline due to scheduling overhead. However, if the set of channels are not able to transmit the SIUs upon successful delivery of the schedule due to their current load, then these SIUs' deadlines are going to be missed anyway even if they are assigned immediately. In this situation, the value of the time quantum is extended to maximize the quality of the schedule.

4. Performance evaluation

A comparative study in the context of a simulated set of experiments evaluates the performance of the proposed algorithm. The candidate algorithms are the time-constrained search algorithm discussed in this paper and the forward algorithm proposed in [1]. Two versions of the search algorithm were implemented. The first version labelled SA in the performance results is a pure search algorithm, which treats all dead-ends as hard dead-ends. The second version of the algorithm labelled SA-SD employs the selective dropping technique which differentiates between between various types of traffic to comply with the user's QoP requirements. The motivation behind the implementation of the two versions is to evaluate the effect of the selective dropping technique employed by the search algorithm. The other candidate algorithm, namely the forward algorithm and labelled in the figure FA works as follows. SIUs are first ordered in an increasing order of their playout deadlines. Each SIU is then scheduled on the channel that results in the earliest transmission time. The forward algorithm belongs to the best effort class of algorithms by opposition to our

algorithm, which is feasibility-based, and feasibility checks are performed at the server side. The experiment results were obtained following 10 runs of the simulated algorithms. The mean of the 10 runs was plotted in Figure 4.

The problem set consists in a set S of Poisson-based distributed SIUs. The performance metric is the percentage of SIUs reaching the client side before their playout deadlines. Several experiments were conducted but due to space limitation, we show below the result of a single experiment showing the rate of successfully transmitted SIUs at the y-axis versus the deadline factor at the x-axis which low values reflect tight SIUs' deadlines whereas high values correspond to loose SIUs' deadlines. SIUs' sizes are uniformly distributed between 0.1 Kbytes and 6 KBytes. The deadline, d_i, of SIU_i is uniformly distributed within the interval (End_i, D_{max}) measured in milliseconds, where End_i is the worst-case transmission time of SIU_i (i.e. assuming it uses the channel with the lowest bandwidth). D_{max} is calculated as follows: D_{max}= Deadline_Factor × End_i where Deadline_Factor is a parameter in our experiment that controls the degree of laxity in SIUs' deadlines. Larger Deadline_Factor represents larger slack times, whereas small Deadline_Factor represents tight deadlines. The network resources consist in four channels with bandwidth 1, 2, 3 and 4 Mbps respectively. Finally, note that in our experiment, the scheduling cost for the Forward algorithm is set to 0, whereas the search algorithm is penalized by its incurred time-complexity which is however maintained under control as discussed earlier in this paper.

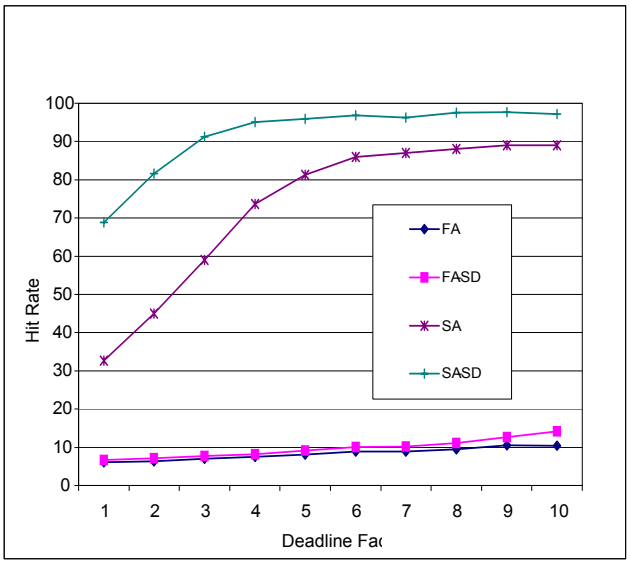


Figure 4. Transmission rate(%) vs. Deadline Factor on 4 network channels with bandwidth 1, 2, 3 and 4 respectively.

As shown in Figure 4, the proposed search-based algorithm outperforms the forward algorithm in terms of SIUs' deadline hit-rate by as much as 80% at loose deadlines. This means that the search-based algorithm takes better advantage of deadlines looseness in maximizing the transmission rate of SIUs. Such performance gain by the search algorithm is attributed to the fact that the search algorithm checks for deadline feasibility in the schedule whereas the Forward algorithm is only based on channel availability times. Also, the forward algorithm does not scale-up performance as much as the search based algorithm does when equipped the selective dropping mechanism. This is attributed to the fact that SA-SD algorithm progresses faster towards a feasible schedule than SA algorithm since in SA-SD, backtracking is attempted only when hard dead-end are encountered during the search process, whereas SA backtracks to the previous search level whenever an SIU cannot be scheduled on any of the available channels. Such backtracking, delays the scheduler from moving deeper in the search space towards a leaf node, which reflects a complete feasible schedule. The margin of performance of SA-SD over SA is about 40%. This is indicative that the selective dropping technique which makes use of the users' QoP requirements is employed efficiently by the search algorithm especially at tight deadlines, where backtracking situations are often encountered.

5. Conclusion

In this paper, a dynamic non-pre-emptive real-time scheduling algorithm for high-level multimedia synchronisation on broadband networks has been proposed. The algorithm runs at the server side above the ATM layers to negotiate the transmission of media objects composing multimedia documents so that they reach the client side before their playout deadlines.

6. References

[1] Shahab Baqai, M. Farrukh Khan, Miae Woo, Seiichi Shinkai, Ashfaq Khokhar, and Arif Ghafoor, *"Quality-Based Evaluation of Multimedia Synchronisation Protocols for Distributed Multimedia Information Systems"*, IEEE Journal on Selected Areas in Communications, Vol. 14, No. 7, pp1388-1403, 1996.

[2] Atif Y, "*Dynamic Load-Assignment in Distributed Memory Multiprocessors*", International Journal of High-Speed Computing, vol. 10, No. 1, pp. 83-113, 1999.

Synchronous Time Division Internet for Time-Critical Communication Services

Takahiro Yakoh
Department of System Design Engineering
Faculty of Science and Technology, Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama 223-8522 Japan
yakoh@sd.keio.ac.jp

Abstract

This article proposes a new framework of the Internet implementation for the purpose to be able to minimize the transmission delay time of time-critical packets. The framework consists of new routers which make possible to work both of two modes (shared mode and exclusive mode) on a same network media, and a protocol to synchronize the timing to change the active mode of routers. Usually, the router acts as a conventional router of the Internet in the shared mode. On the other hand, in the exclusive mode, the router acts as a repeater and transfers packets without any rewriting of packet headers. By changing appropriate routers to the exclusive mode, a certain source and destination pair in the network can be connected as if they are connected directly with an isolated LAN. The timing of the mode changing is managed by a QoS (Quality of Service) management protocol so as to change all of the appropriate routers to the exclusive mode synchronously. As the results, the Internet can provide reserved time-critical communication services by providing synchronous time-division slots of exclusive mode to the connection.

1. Introduction

Recently, fine-grain distributed control has become possible to realize. For example, the author showed a bilateral control system which two homogeneous robots move synchronously with communicating each other in every $1ms$ control loop[1]. The system used an isolated Ethernet LAN because multi-hop network can not ensure the reachability of packets in a certain transmission delay time.

To resolve this restriction and to realize geographically distributed control systems, this article proposes a new framework of the Internet called *synchronous time division Internet (STDI)*. STDI is based on *pros* and *cons* of conventional network hardware categorized as follows:

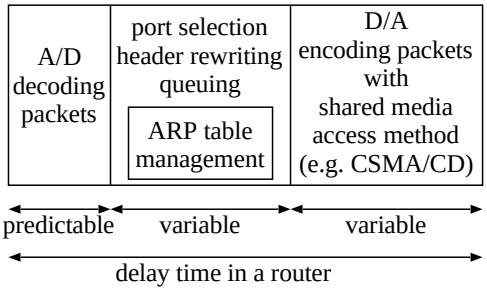
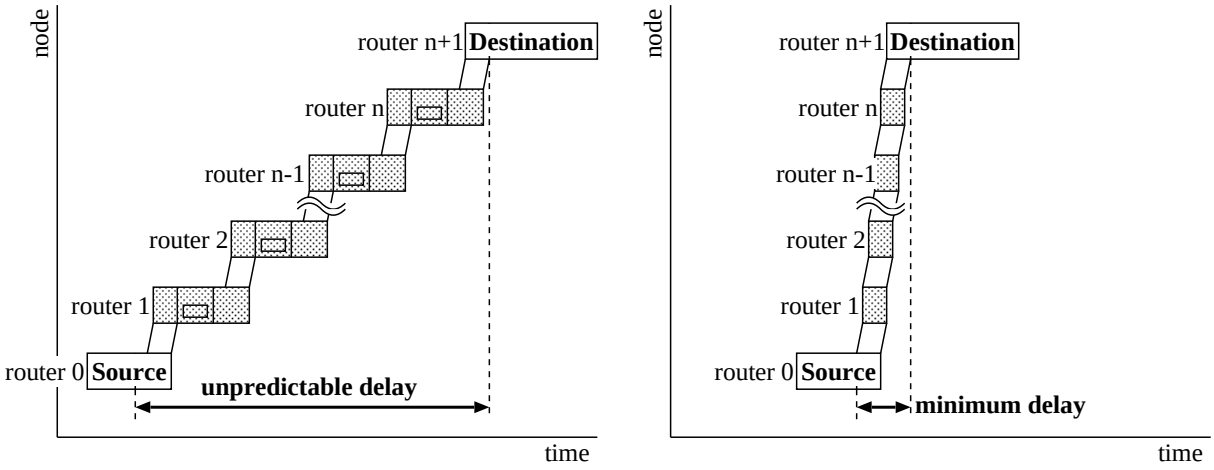


Figure 1. Processing Time in a Conventional Router

router, bridge They can forward packets for infinite hops. But they spend unpredictable long delay time (Fig.1). Store-and-forward type switch is included in this category.

dumb hub, repeater They spend only static short delay time. But they can forward packets only for a few hops because of the distortion and the attenuation of signals. Cut-through type switch is included in this category.

To make the best use of the advantages of both categories, a new packet forwarding hardware, called *synchronous hybrid router (SHR)*, is introduced. Also, to minimize the transmission delay time for time-critical packets, appropriate SHRs *are changed synchronously* to act as repeaters when time-critical packets are transmitted (exclusive mode), and to act as conventional routers in other term (shared mode). The effect of the proposed two technologies is expected as Fig.2. In shared mode, all of the routers cause unpredictable transmission delay (in Fig.2(a)). Many of conventional Internet applications, including TCP/IP flow control algorithms and streaming technologies, are designed as to adapt for the unpredictability. On the contrary,



(a) in Shared Mode (b) in Exclusive Mode

Figure 2. Transmission Delay in Multi-hop Network

in exclusive mode, since all of routers and network media are reserved for a source node to send time-critical packets, any media access methods are not required at all. So the packets sent from a source node can reach to a destination node exactly after the minimum delay (in Fig.2(b)).

2. Synchronous Hybrid Router

A SHR is a packet router to be designed as to achieve the minimum delay from the source node to the destination node of time-critical communications. At the same time, the SHR have to be designed to have an interoperability with conventional IP routers. To meet both of these requirements, it is designed as a combination of three parts: a router, a repeater and a synchronous changer. The router part works as a conventional router. It is unavoidable that the delay of a packet forwarding is unpredictable and variable in case of using this part. On the other hand, the delay in case of using the repeater part, described below, can be fixed or predictable. The synchronous changer part selects one of above two parts to work, i.e., the router part is activated in shared mode and the repeater part is activate in exclusive mode. The timing of the changing is synchronized with all of SHRs by a network resource management system likes RSVP[2].

2.1. Fundamental Design of Repeater Part

If all of network resources are reserved at the time slice (described in 3), the media access method such as CSMA/CD is unnecessary in repeater part. So it is possible for SHR to forward a packet from the input port to the output port in the minimal delay time by just repeating input signal to output ports like a conventional repeater. In this case, the delay time ${}^{rep}D_i$ at a router i in repeating a S bytes of packet is fixed although the signal can be repeated only if the bit rate of input port R^{in} is same as that of output port R^{out}.

$$ {}^{rep}D_i = D_i^{decode} + D_i^{rep} + D_i^{encode} = D_i^{rep} + 2\frac{S}{R^{out}} \quad (1) $$

Where D_r^{rep} is repeating overhead and can easily be designed as a constant (see Fig.1). So the total transmission delay D^{total} is given as a predictable value from the size of a packet.

$$ {}^{rep}D^{total} = \sum_{i=0}^{n} {}^{rep}D_i = \sum_{i=0}^{n} D_i^{rep} + \sum_{i=0}^{n} M_i + \frac{S}{R^{out}} \quad (2) $$

$$ = {}^{rep}AS + {}^{rep}B \quad (3) $$

Here D_i^{encode} and D_{i+1}^{decode} are overlap with shift M_i which is a transmission delay in network media between router i and $i+1$. So the total delay can be calculated easily from the first order function of the packet size. The second and the third terms in Eqn.(2) are exist even in the delay with a fiber cable. So only $\sum D_i^{rep}$ (typically several ns) is the overhead of exclusive mode of STDI.

Please note that the SHR have to be equipped for a signal shaper to compensate the distortion and the attenuation of the input signal so as to make packets to hop many SHRs. Even so, the repeater part can be design as to make D_i^{rep} constant. Also note that there is a precondition that the bit rates of all network media are the same.

2.2. Extended Design of Repeater Part

To resolve the precondition, the repeater part should be extended. If R^{in} is differ from R^{out} then a store-and-forward

type packet forwarding have to be used. Usually the delay time in store-and-forward type switch ${}^{saf}D_i$ is not predictable because of queue processing and media access processing.

$$ {}^{saf}D_i = D_i^{decode} + D_i^{queue} + D_i^{encode} + D_i^{ma} \quad (4) $$

Where D_i^{ma} is the media access processing time (see Fig.1). D_i^{queue} depends on the load of the router i and D_i^{ma} depends on the condition of the network media between router i and $i+1$. But since the network resources are reserved to use and d the router process only the time-critical packet with a SHR, D_i^{ma} can be zero and D_i^{queue} can be fixed. So the delay time can be predictable with the following first order equation.

$$ {}^{saf}D_i = \frac{S}{R^{in}} + D_i^{queue} + \frac{S}{R^{out}} \quad (5) $$

$$ {}^{saf}D^{total} = \sum_{i=0}^{n} {}^{saf}D_i \quad (6) $$

$$ = \sum_{i=0}^{n} \left(D_i^{queue} + \frac{S}{R_i^{out}} + M_i \right) \quad (7) $$

$$ = {}^{saf}A S + {}^{saf}B \quad (8) $$

As the results, there are two ways to implement a repeater part, i.e., the fundamental repeater type and store-and-forward type without media access method. Both of them can work as a SHR because the delay time of the repeater part is successfully bounded as Eqn.(3) and (8). So the total delay time from the source to the destination can also be bounded as the first order function of the packet size.

Obviously ${}^{saf}A \gg {}^{rep}A$ and ${}^{saf}B \gg {}^{rep}B$, so the fundamental repeater type behavior is much better than the store-and-forward type one. SHR i should be designed to support both of repeater types and to use fundamental repeater type if R_i^{in} and R_i^{out} is the same.

2.3. MAC Address Aliasing and Port Selection

In general, a router have to select the output port and rewrite the source and destination MAC addresses of a forwarding packet along with an ARP table in the router because physical layer of network refer MAC addresses to receive or forward a packet. Fig.3 shows the overview of MAC address rewriting where MAC_i^{in} and MAC_i^{out} are MAC addresses of input port and output port on router i. SHRs in shared mode also have to select the output port and rewrite MAC addresses as same as the conventional routers.

On the other hand, SHRs in exclusive mode should not refer its ARP table because referencing the table may cause unpredictable delay. Furthermore, it is better to omit packet rewriting to shorten D^{rep} or D^{queue} and to simplify its implementation (Fig.4). To omit packet rewriting, the following two questions arise.

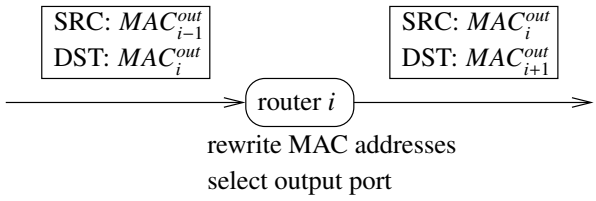


Figure 3. MAC Address Rewriting in router i

- How to receive packets destinate to another MAC address
- How to select the output port

To answer these questions, all of SHRs have to know the MAC addresses of the source and destination nodes and the route of the time-critical communication in advance. So it is necessary that these information is included into the resource management protocol.

Furthermore, if an intelligent switch exists between two SHRs, the switch have to learn the MAC addresses of the source and the destination of time-critical communications even though these nodes do not connected to the switch directly. So explicit or implicit teaching is required for all of these switches.

3. Synchronous Time Slot Reservation Method

To ensure certain QoS between the source node and the destination node of a time-critical communication, a resource reservation system is necessary. In STDI, all of routers and network media, which consists a route from the source to the destination of a time-critical communication, have to be reserved only when the time-critical packets are sent (the minimum delay term in Fig.2(b)). In other words, time slice of network route is defined as a resource in STDI, although bandwidth, measured in relative long period, is defined as a resource.

The most typical applications which require a time-critical communication are closed loop control systems. For example, bilateral robot system[1] is consists of two nodes which communicate with each other every $1ms$ period. Each packet size is relative small (less than 100byte) so one reservation term is short (less than 1μs). In general, communicating control systems require a short communication periodically.

So SHR is designed to support time slot reservation for all network lines and nodes. For the above example, time slot of 1μs per $1ms$ should be reserved to support the application. During the reserved term, all of SHRs are changed to an exclusive mode. On the contrary, they are changed to the shared mode during the rest term (i.e. 999μs per $1ms$). Fig.5 show the typical example of mode transition cycle

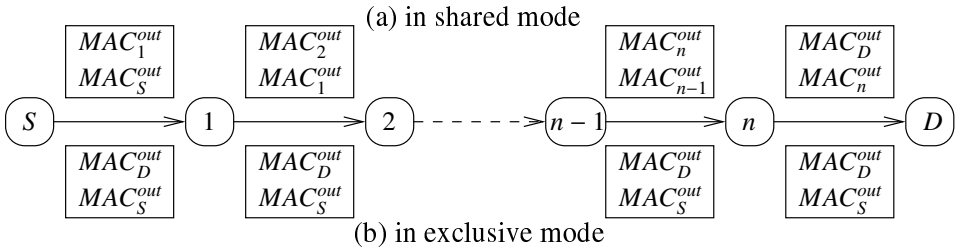


Figure 4. MAC addresses in each packet

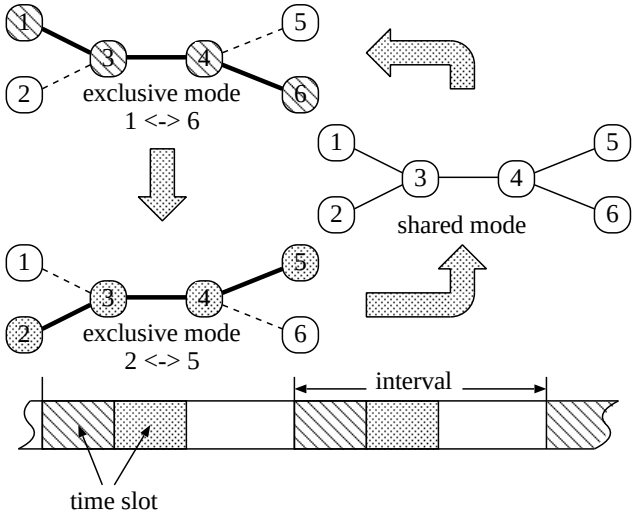


Figure 5. An Example of Mode Transition Cycle

when two time-critical communication exist in a network.

3.1. Synchronization of clocks

To achieve the time slot reservation as described above, precise adjustment of clocks in every SHRs is indispensable. NTP[3] provides adjustment method of computers, but 5ms of offset is a limit of NTP in normal usage. Horauer and Höller achieved the accuracy of synchronization in the range below 100μs with some hardware support [4]. NTP over synchronous time slot of STDI is expected to achieve high accuracy of synchronization.

4 Applications

Although STDI is designed for the purpose to realize time-critical communications through the Internet infrastructure, it is possible to apply STDI to various applications. The followings are

IP Telephony Voice over the Internet is one of the new application of the Internet. In general speaking, it is in the relation of the trade-off with the delay and the stability. STDI can provide both of short delay and high stability.

Multiplexing Virtual Private Networks When the exclusive mode is assigned to the arbitrary number of nodes and MAC address aliasing turns off, these node can be connected as if there is a private network.

5. Conclusion and Future Works

This article proposed STDI, a new framework of the Internet, to minimize the transmission delay time for time-critical packets. SHR, a multi-functional IP router, was designed to realize STDI. This article showed the transmission delay between distanced two nodes can be successfully bounded with STDI and SHR. Especially when all of network media has the same bit rate, the delay is near to that of a lease line.

References

[1] T. Yakoh, H. Sato and T. Aoyama, *Fine Real-Time Processing in Distributed Systems*, WFCS, 2000.

[2] R. Braden Ed., *Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification*, RFC-2205, IETF, 1997.

[3] D. L. Mills, *Network Time Protocol Specification*, RFC-1305, IETF, 1992.

[4] M. Horauer and R. Höller, *Integration of high accurate Clock Synchronization into Ethernet-based Distributed Systems*, International Conference on Advances in Infrastructure for e-Businnes, e-Eudcation, e-Science and e-Mediscine on the Internet, 2002.

CONVERGENCE

Ole Brun Madsen, J. Dalsgaard Nielsen and Henrik Schiøler
Aalborg University, Institute for Electronic Systems
obm,jdn,henrik@control.auc.dk

Abstract

Convergence trends between the WAN Internet area, characterized by best effort service provision, and the real time LAN domain, with requirements for guaranteed services, are identified and discussed. A bilateral evolution is identified, where typical bulk service applications from WAN, such as multi media, migrate into the RT-LAN domain along with the need for extensible and easily maintainable technology, demanded by such applications, to coexist with QoS demanding applications on a common platform. Meanwhile QoS demanding dependable applications find their way out into WAN with the emergence of remote service provision, such as supervision and control of decentralized heating facilities and wind based electrical power production. The reliability issue is addressed from a structural viewpoint, where the concept of Structural QoS (SQoS) is defined to support reliability modelling in communication infrastructures. A graph theoretical approach is presented as an approach to reliability management in complex communication infrastructures. Real life examples are provided and specific problems are presented and discussed. Wireless technologies are discussed as a complement, providing not only mobility and installation ease but also a complementary failure profile.

1. Introduction

It is our claim that the ongoing convergence between WAN and RT-LAN is bilateral, since it holds both the migration of Internet traffic to the RT-LAN area, and the adaptation of the global information system to support dependable applications with high QoS demands. In addition, short-range wireless technologies approach a well-established position, supported by only a few unifying de-facto standards, such as IEEE 802.11b and Bluetooth. This last fact seems to impact the penetration of wireless technologies into dependable areas more than pure technical arguments.

QoS sensitive applications move onto WAN in two categories: non-dependable QoS sensitive applications like multi media and VoIP, and dependable applications like remote security services and remote control. Several studies of the former have been made, dealing with modelling traffic sources and network traffic modelling [1]. This research is mainly concerned with queuing and delay considerations under normal operation mode.

For dependable applications, additional issues regarding system failure remain to be considered. Long term reliability and availability requirements need to be specified. From a provisioning viewpoint, policies for operation under failure, supporting graceful degradation, are to be specified. Provisioning service to dependable applications should be based on pre-negotiated QoS contracts, including guaranteed reliability levels based on thorough reliability modelling, involving the current customer portfolio, failure policies, as well as the underlying network infrastructure. The liberalization of telecom service provision is complicating the process by keeping relevant infrastructure information hidden from the end-to-end service provider. Conversely, multi media services for e.g. surveillance and supervision purposes migrate into traditional control system areas.

Wireless technologies complement their wired counterparts both in the broadband and in the real time LAN domains, i.e. tedious cabling may be avoided and wireless communication obviously facilitates mobility.

Wireless technologies exist in the global infrastructure, ranging from satellite and radio link communication over Fixed Wireless Access (FWA, IEEE 802.16) to a LAN and PAN level with existing short-range technologies like DPRS (EN 301 469-1), WLAN (IEEE 802.11b) or Bluetooth (IEEE 802.15)/[2] and home-RF [3] along with experimental technologies like HIPERLAN (EN 300 652) and Ultra Wide Band (UWB) [4].

Generally, wireless communication technologies suffer from limited bandwidth and less reliable link services. We suggest wireless technologies for improving the overall reliability of an entire communication facility, since failures in wireless links are basically of a different nature than for wired links.

The paper is organized as an introduction to the concept of SQoS along with an overview of the global information infrastructure. This is followed by an example of real time applications migrating to WAN environments, describing a technical network under implementation inside a local Danish municipality. The ATOMOS standard is presented along with the reliability issues behind the standard. Wireless communication protocols are presented and discussed w.r.t. QoS and suggestions for their roles in improving SQoS are put forth. Finally, concluding remarks are given summarising the standpoints presented in the present work.

2. SQoS and Topology

Structural QoS (SQoS) is a novel concept introduced in [5], dealing with QoS parameters primarily related to architecture and structural properties in the infrastructure. SQoS forms the base for support of a variety of services established across the infrastructure, demanded by the communication services. Requirements relate primarily to delay, reliability and capacity issues, and are specific for the individual types of applications using the communication service in question. Explicit knowledge of the inherent topological properties is important in order to provide a viable platform for providing differentiated SQoS.

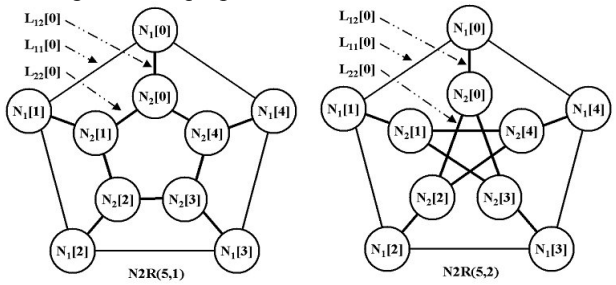
The backbone infrastructure in a WAN network is today typically built as a meshed network interconnecting a number of local fibre optical ring structures, extending into part of the MAN area. The nodal redundancy is obtained either by duplication of equipment or by geographical diversity. Although the backbone structure allows for at least triple connectivity globally, most operators have chosen a service provisioning strategy based on ring structures at this level as well. One reason being, that this choice is supported by automatic restoration algorithms on the optical fibre level, and the fact that the meshed structures in general are build in an ad-hoc manner. The latter is causing a complex time-consuming restoration algorithm, generally unsupported. This is preventing the full utilisation of the inherent potential for higher SQoS. For the support of mass driven demands with low reliability, the ring structure based automatic restoration in the WAN/MAN area is sufficient, given that the last mile access in general only allows for single line connectivity. For higher SQoS demands, automatic restoration cannot compensate for the permanent availability of more physically independent lines.

A straight forward possibility is to combine the wired and the wireless based network in a joint service-provisioning platform in the MAN area, offering at least the potential for dual independent access for the end-user, and to a large extend even more. This is opposed to the actual situation where the radio based access network is seen as a competing technology to the wired networks. The prerequisite is a careful planning of the placement and linking of the base stations in the radio-based networks to avoid dependencies with allocated channels to the end-user in the wired networks.

Applying structuring principles beyond the level of ring network topologies is a promising complementary approach. To avoid the complexity in the meshed networks in general, an approach based on regular graph structures is investigated. By applying methods from graph theory, rather large structures can be handled with a realistic computational effort. For illustration, consider a family N2Rpq of regular degree-3 graphs. N2R(p;q) can be described as two interconnected rings:

Nodes: $\{N_1[i] \cup N_2[i]\}$;
Lines: $\{L_{11}[i] \cup L_{22}[i] \cup L_{12}[i]\}$
$L_{11}[i]$: Line $(N_1[i], N_1[i+1 \bmod p])$
$L_{22}[i]$: Line $(N_2[i], N_2[i+q \bmod p])$
$L_{12}[i]$: Line $(N_1[i], N_2[i])$

$0 < i \leq p$; p and q are integers without common prime factors, $p>2$; $1 < q < p/2$.



N2R(p;1) correspond to the double ring architecture. N2R(5;2) correspond to the Petersen graph. The N2R(p;q) family allows for the provisioning of a static routing scheme, with global knowledge of distances and paths across the network and therefore also a predictable and specified base for delay and reliability calculations. The highest SQoS demand that can be provided on a global scale in N2R(p;q) is a triple set of dedicated independent connections between any two points with a predictable upper limit for the delay variations between the 3 connections. For any specific value of p an optimal value of q can be calculated with respect to minimizing the resource consumption for the service provisioning.

In [5] and [6] a detailed analysis of networks with a SQoS potential is provided, in order to investigate the viability of a systematic approach for a specific representative area. As a special case, a Swedish model [7] is analysed. This model is based on a logical square structure with local recursive refinements.

3. Global Infrastructure

The global infrastructure can be described as a huge population of Personal Area Networks (PAN's) or field level busses, a variety of customer premises networks (LAN's), a set of copper or radio based access networks (Man's) interlinked by a set of fibre optical based backbone networks (WAN's).

The IT infrastructure has evolved from a set of more or less vertically integrated networks dedicated to specific application service types, moving towards an interworking set of horizontally integrated and converging infrastructure platforms common for literally all services. The complete digitalisation of voice and video services making this evolution possible forms the base for convergence. It also opens for the potential of bringing data and telemetric services with strong QoS requests into the MAN and WAN area, previously limited to the LAN area due to the

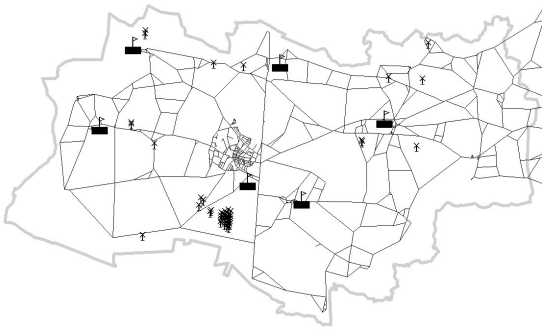
relatively high costs in dedicated leased line based MAN and WAN sub-infrastructures.

WAN. The backbone networks (WAN's) are typically built as interconnected sets of optical rings with SDH and ATM equipment. The structure originally followed the structure in the telephony networks, but is evolving more and more in direction of ad hoc networks, not following an overall commonly agreed upon architecture. This makes the possibility of establishing well-specified global infrastructure services increasingly difficult.

Under existing policies of a liberalised telecommunication marked this could lead to a replication of the existing architectures as well as an infrastructure only covering the most profitable areas in the densest populated areas. That would prevent the possibility of extending services with stronger SQoS demands to the wide area on a global scale and thereby missing a large-scale new market potential. This situation has now finally been recognized by both industry and the public authorities. EU initiatives are in progress for the coming period to find solutions to this problem.

As part of a Danish national program [6] for promoting a modern IT based society, an investigation has been made in the analysis and design of a new access network architecture in the local municipalities in the Northern Jutland region (6000 km^2 and 600.000 inh) taking advantage of the opportunities in the situation. The outcome is a strategic long-term development plan for the area. An important element in the plan is a new fibre optical MAN with FTTH and carrying an identified set of virtual networks reflecting the variety of SQoS demands implemented in a common infrastructure.

The most ambitious part of the plan is a virtual technical support network that allows for distributed real time control application throughout the region with committed global delay and reliability parameters.



One of the participating municipalities (300 km^2 and 25.000 inh) has been actively contributing to investigate and implement elements of a new MAN infrastructure along these lines. The first advanced application target is a shared surveillance centre for seven smaller heating and electrical power plants distributed over the municipality.

The long-term goal for this application is a shared control and operational centre for real time coordinated production regulation amongst the plants, including a relatively large amount of electricity producing windmills with a reliability corresponding to the present on-site quality.

MAN. The most widespread types of copper based access networks (MAN area) are based on the existing telephony access networks or cable TV distribution networks, with no possibility for redundancy in general. The wireless networks complete the picture with mobile as well as fixed access. The explosive evolution in access capacity calls for an implementation of an entirely new access infrastructure based on optical fibres to the home and complemented with next generation wireless technology.

LAN & PAN. For production plants of significant size, a number of available communication technologies exists. Commercially a limited set like devicenet [8] , profibus (DIN 19245) and EcheLON (ANSI/EIA 709.1) seem to dominate. In [9] it is argued that an automation LAN should cover 3 separate areas; a top administrative level, a mid real time level and a hard real time field level, and should be technologically separated accordingly in order to support specific QoS levels characteristic to each level. Other arguments, such as flexibility, ease of maintenance and SQoS issues, point away from such a separation. In the EU funded ATOMOS I project [11] a communication standard for marine automation was based on a single technology, ARCnet (ATA/ANSI 878.1), covering all three levels. Reliability was at that point accounted for in the shape of replication at a segment level, i.e. double NICS and cables comparable to class 1-2 configuration in TTP [9] . From an SQoS viewpoint two critical points associated with such an approach may be identified; requirements for physical cabling remains unspecified leading to possible dependent cable damage, and structural reliability issues are unspecified outside a segment level. Overall SQoS considerations would lead to architectures where segments act as vertices in topologies with well specified SQoS properties.

In unified network architectures, all three of the above traffic categories should coexist on a single media. Trends towards integrated "intelligent" components, e.g. pumps with http interface, facilitating "thin clients" or AC motors with integrated velocity controls naturally pulls the traffic profile away from the hard real time field level towards a mid real time level with modest RT requirements. Deterministic network calculus [1] from real time analysis in ATM networks is applied in [10] to bursty real time traffic in the ATOMOS architecture.

Evolution in standards and technology has brought wireless communication to a significantly larger application area lately. Standards and technological implementations exist for a variety of applications ranging from global

satellite communication to local communications in the centimetre range with Bluetooth power level 1. In between, we find radio link, wireless access, mobile telephony, wireless LAN, and wireless PAN technologies. Most of the mentioned technologies are primarily intended to support merely bandwidth demanding applications like voice and multi media. It remains an open question, which role such technologies are to play w.r.t. dependable and delay sensitive applications like distributed feedback control. Wireless transmission suffers from notoriously high bit error rates, i.e. 1E-5 to 1E-3 compared to 1E-9 in wired links. However, with redundancy coding like the 1/3 FEC, Bluetooth supports a rate of 800 16-bytes frames pr. Second at an error rate (FER) within 4E-8 to 4E-4, when used as a point-to-point wireless connection. Thus at a field level Bluetooth may act as a sensor/actuator bus at ranges up to 10 metres. However, Bluetooth power consumption may be far too large when battery life times above 2 years are required. Superior bandwidths at all ranges are reported from UWB experiments at power levels 13 dB below Bluetooth. Commercially available solutions are still far above tolerable power levels for battery-powered applications.

At an increased range level, consider remote and distributed control of decentralised heating facilities and windmill parks. Wind mills are typically located in rural areas so wired communication lines are bound to pass fields used for agricultural purposes, increasing the risk of line break. For such a situation FWA technologies at nominal bandwidths up to 2 Mbps and ranges from 1-10 km seem appropriate as backup lines or hot stand by. At shorter range, and considering the ATOMOS network, the intended increase in reliability may be significantly reduced in fire situations where replicated cables are likely to be damaged simultaneously. Wireless backup communications should be considered at an early stage in an integrated SQoS design process to secure communications vital to dependability in e.g. fire situations. The IEEE 802.11b standard seems appropriate for such a task considering both range and bandwidth.

4. Concluding remarks

In the picture of a converging global infrastructure, some of the problem areas have been highlighted. The introduction of an SQoS concept points to new systematic approaches in order to create support for dependent applications with high reliability service demands.

In the LAN and PAN environment, the technologies are ready, and the challenge lies in improved architectures encompassing the system integration process. Opposed to the actual WAN situation, the fast increasing density of interworking wired and wireless components leaves plenty of room for establishing sub-network topologies with high SQoS potential. This, on the other hand, gives problems in maintaining SQoS based substructures within mobile components, as the high potential for sub-structuring at the same time calls for improved strategies for the selection and maintenance algorithms in the wireless line and nodal space.

A key focus point in the next period should be the MAN environment, as the component in the global infrastructure with lowest SQoS potential. Based on the explosive capacity-driven demand, we are confronted with the need for a complete re-implementation with fibre optical technology, complemented with new wireless technology. This situation gives a historical chance to create an improved global infrastructure with an overall high SQoS potential extended not only to the MAN level, but also all the way to the WAN level. The density of branching points in the new fibre optical structures will increase from an average of app. 0,04 to 1 per sq km. By careful planning of the MAN it will leave room for an extremely low cost complementary set of lines in the WAN back-bones.

References

[1] R. L. Cruz, “A Calculus for Network Delay Part I: Network Elements in Isolation”, IEEE trans. on Information Theory, 1991

[2] Bluetooth Special Interest Group, "Specification of the Bluetooth System”, available at http://www.bluetooth.com

[3] http://www.homerf.org /data/tech/HomeRF_QoS_whitepaper.pdf

[4] http://www.palowireless.com/uwb/turorials.aps

[5] The Structural Impact on Quality of Service parameters http://www.control.auc.dk/sqos/

[6] The Digital North Denmark project http://thedigitalnorthdenmark.com/index.php/m/180/

[7] General guide to a future-proof IT infrastructure, Report 37/2001, the Swedish ICT Commission.

[8] Allen-Bradley Communication Networks Overview http//www.ab.com/catalogs/b113/comm/overview.html .

[9] H. Kopetz, "Real-Time Systems, Design Principles for Distributed Embedded Applications", Kluwer Academic Publishers, Massachusetts, 1997

[10] H. Schioler , N. Nielsen , J. Nielsen and N. Jørgensen, „Worst case Queue Length Estimation in Networks of Multiple Token Bus Segments”, Proceedings of ISCA PDCS98, 1998

[11] M. Granum-Jensen and T. N. Hansen and N. Jørgensen and J. F. D. Nielsen and K. M. Nielsen, Technical report AT2312-2, EU-DG7, 1993