By Salem NASRI and Brahim Maaref,
Ecole Nationale d'Ingénieurs de Monastir,
and LSR-IMAG.

Implementation and Performances Evaluation of Manufacturing Communication System Over ATM

The performance and the architecture of a network interconnecting diverse functions of a Computer Integrated Manufacturing (CIM) are of great importance. Indeed, the majority of these functions have time-critical constraints that must be considered when designing such a network. In this paper, an architecture based on the OSI reference model is presented, along with service elements and protocols, which match the requirements for these constraints. Our architecture is based on ATM (Asynchronous Transfer Mode) and MMS (Manufacturing Message Specification) application layer protocol. In this paper, we study the implementation of an MMS/ATM architecture. Then we evaluate the performance of this implementation from an external and internal point of view.
To increase performance, we offload high overhead MMS functions to a hardware co-processor. Software parts of the architecture, especially the buffer management have been carefully implemented. As a result, we have achieved a better performance than in previously reported MMS implementations.

OBJECTIVES

Performance can be evaluated in different ways: one-way delay, response time for request-response protocols, throughput in bytes or in requests per second etc. In case of the MMS protocol [1,2], performance is assessed in terms of response time for MMS request-response services. The purpose of this paper is to evaluate the performance of an Ada implementation of the MMS protocol over an ATM network [3]. This evaluation is made in details, adopting both an external point of view, for requests made to the application layer, and an internal point of view, to understand how the time is spent in the different parts of the communication system.

We study, in detail, the transmission delay caused by the execution of the MMS protocol layers. For demanding environments, it may be sufficient to provide high-performance services with low response times and high bandwidth. Obviously, increasing performance does not make a protocol real-time, but it may be a first step towards this goal. This approach has been taken by the Mini-MAP architecture [4], which enhances MMS performance by reducing the number of protocol layers to three. We follow this approach by proposing a high-performance MMS architecture that achieves even better performance.

Our architecture is based on three design choices:

1. high-speed communication fiber-based ATM network providing 155 Mbps ATM and supporting quality of service (QoS),
2. offloading of high overhead functions of the MMS protocol to a hardware co-processor,
3. careful implementation of software parts of the MMS protocol.

In this way, we obtain better performance than those previously reported MMS implementations. High performance and predictability of low-level ATM communications are the basic features on which relies a real-time implementation of MMS.

INTRODUCTION

MMS overview

MMS is an OSI application layer protocol that enables remote applications (called clients) to control and supervise various heterogeneous industrial devices (called servers). MMS is a part of the MAP architecture [4]. It defines only two aspects of communication in an industrial network. The first one is the concept of a Virtual Manufacturing Device (VMD), which essentially presents an abstract view of a physical device, and the second one covers objects, services and protocols used to support communication between such abstract devices [1,2]. The VMD hides the complexity of the real industrial device and provides a common understanding of all such devices. MMS defines a large set of 86 services operating on 19 classes of objects.

MMS is becoming widely accepted for open communication between heterogeneous devices in many areas, not even limited to manufacturing. For example, it might be used in a space station. In the initial development phase of Space Station Freedom, a major effort was launched to establish comprehensive information management architecture for operation of the spacecraft. A number of innovative concepts were developed during that period of time. Early design efforts determined that a messaging service on top of a packet transport service would greatly enhance the operation of the Space Station. However, the 1990s

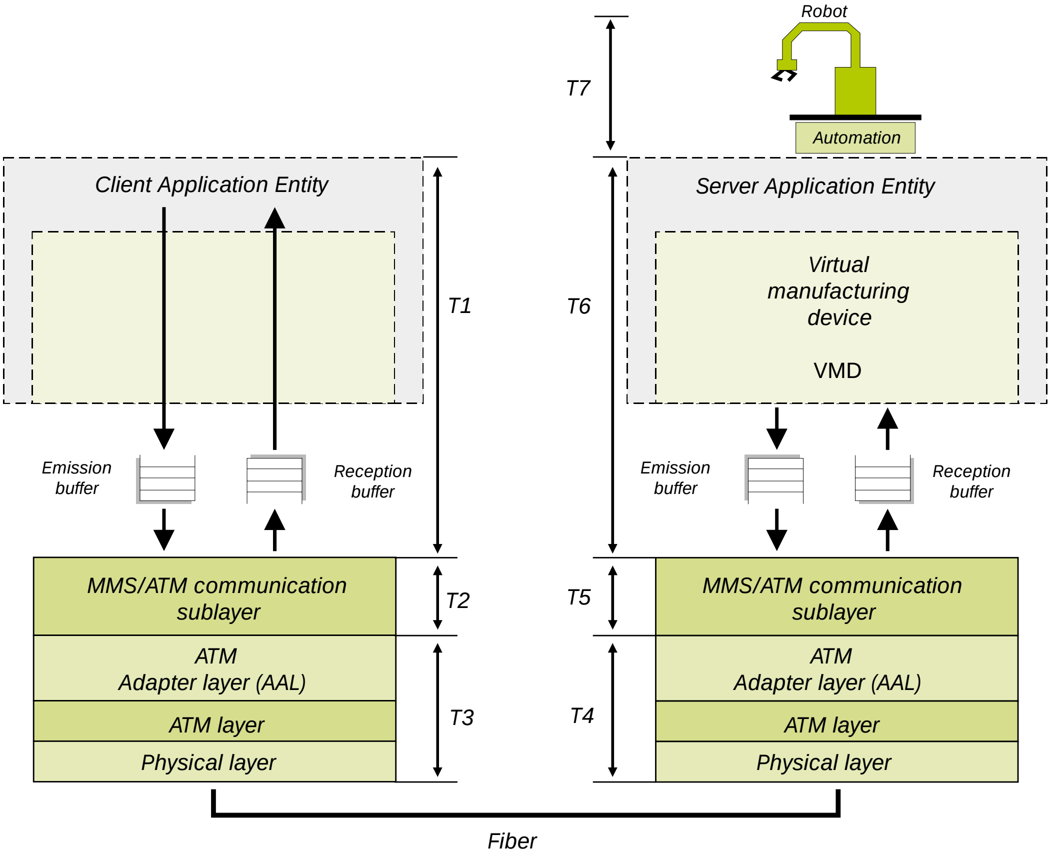

Figure 1. MMS/ATM Communication System Architecture.

have seen Space Station Freedom scaled back from its original scope and many of these innovative concepts have been dropped. Despite the re-direction of Space Station, a small group of interested parties continued to meet and discuss the challenges of a comprehensive command and control system for space missions. The Space Project Mission Operation Control Architecture (SuperMOCA) has emerged from those discussions as an effort to develop an "open specification" for civil and military space projects. The Manufacturing Message Specification has been selected to develop the SuperMOCA [5].

We can also cite another example of MMS use. The Telecontrol Application Service Element (TASE.2) protocol (also known as Inter-Control Center Communications Protocol, ICCP) allows for data exchange over Wide Area Networks (WANs) between a utility control center and other control centers, other utilities, power pools, regional control centers, and Non-Utility Generators. It defines a standardized method of using MMS services to implement the exchange of data [6].

ATM Plant Control Network

ATM networks are expected to become the next generation of digital communication networks, and provide high-quality and flexible communication services both for wide-area information transportation over B-ISDNs and for the basic infrastructure of computer networks and intra-device interconnections [7, 8]. Two key techniques are used for ATM networks to achieve the above goal: asynchronous multiplexing and fast packet switching. Asynchronous multiplexing enhances the efficiency of a network to transport varying bit-rate traffic since many users can dynamically share a transmission link. Fast packet switching reduces the message-transmission delay by implementing switching functions in hardware.

ATM is an emerging technology used to provide integrated services for high-speed digital communication networks. Its ability to support high bandwidth, high reliability and guaranteed quality of service communication makes it an ideal candidate for the next generation of plant control networks. In a research project, Mitsubishi Electric Research Laboratories (MERL) is collaborating with other research groups to provide advanced traffic management and fault-tolerant functions in Mitsubishi Electric's next generation Real-Time ATM Plant Control Network (RTPCN). From these groups, we can cite: Mitsubishi Electric's Information Technology R&D Center (ITC), Industrial Electronics and Systems Laboratory (IESL), Power and Industrial Systems Center (PISC) and the University of Michigan (UM), [9].

AN MMS/ATM COMMUNICATION SYSTEM ARCHITECTURE

The goal of our MMS architecture is to provide a new communication system in an industrial environment. To achieve this goal, we have chosen to use ATM and to reduce the number of layers by interfacing the MMS

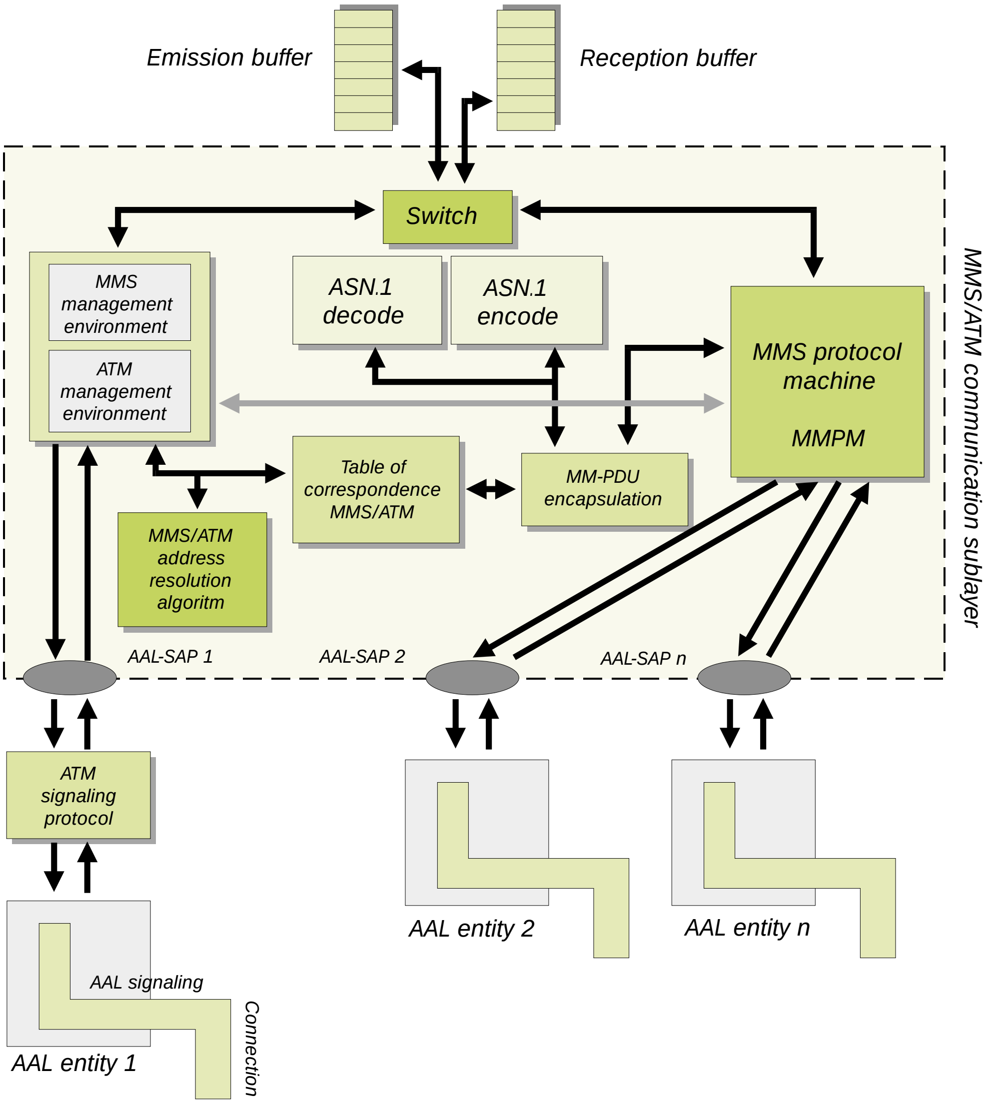

Figure 2. Communication Sublayer Specification.

application layer directly to the ATM adaptation layer.

Our MMS architecture consists of four layers (Figure 1):

1. the MMS application layer,
2. the ATM adaptation layer (AAL),
3. the ATM layer and
4. the physical layer.

The lower three layers conform to the ATM reference model and are implemented on our communication board. The MMS application layer is implemented as an Ada program on a host processor.

The MMS application layer

We have split the MMS application layer into two parts: MMS Application Entities, MAE (client and server) and MMS/ATM communication sublayer (Figure 1). The Client Application Entity presents the MMS primitives to applications and communicates with the Server Application Entity (VMD) on a remote device via the MMS/ATM communication sublayer. The MMS application layer in our architecture is similar to the one used in the Mini-MAP architecture. The Mini-MAP application layer offers services of a reduced subset of OSI application service elements [10]. The application entities communicate with the lower layers by means of two buffers: the Emission Buffer and the Reception Buffer.

We describe below the lower communication layers implemented on our ATM communication board.

Lower Communication Layers

An important sublayer in the ATM stack is the ATM adaptation layer (AAL). The function of the AAL is to convert and map information from the higher layers into the ATM layer (and vice versa). AAL functions are arranged into two sub-layers, the Segmentation And Reassembly (SAR) sublayer and the Convergence Sublayer (CS) [3].

The AAL layer provides two access interfaces: the user access interface and the signaling access interface. The signaling access interface allows an application entity to initialize the ATM board as well as to establish, release, and abort a connection. The user access interface provides primitives for creating and destructing ATM ports that are used to send or receive data.

The ATM layer provides a transparent cell switching service to the AAL layer. Our board contains an embedded 4 by 4 ports ATM switch. It is the principal component of our communication system. Based on the values of VCI (Virtual Channel Identifier) and VPI (Virtual Path Identifier) in the cell header, it switches incoming cells to the next ATM switch.

MMS/ATM Communication Sublayer Specification

This section presents the main contribution of our implementation of the MMS protocol directly on the top of the AAL layer. Our contribution defines, essentially, the interface between the MMS and the AAL layer. This interface is called MMS/ATM Communication Sublayer and its architecture is presented in figure 2. This sublayer allows the adaptation between MMS Application Entities (MAEs) and the AAL layer.

The main functions performed in the MMS/ATM communication sublayer are described in the following subsections.

The MMS Protocol Machine (MMPM)

The MMPM controls the exchanges between the communicating entities. The MMS protocol is executed by an instance of the MMPM state machine, which must ensure the link between MAEs and the AAL layer. This MMPM converts the local syntax of MMS primitives and data to the transfer syntax (encode/decode functions). It is different from previously proposed MMS implementations.

The client MMPM state machine performs two actions. Action 1 is the search of the MMS association identifier that corresponds to the AAL connection identifier and the encapsulation the MMS-PDU to an AAL-PDU. However, Action2 is the search of the AAL connection identifier that is associated to the MMS association identifier and the encapsulation of an AAL-PDU to an MMS-PDU.

The MMS-PDU encapsulation

In this section, we describe the encapsulation of an MMS Protocol Data Unit (MMS-PDU) before emission and after reception. There are one emission and one reception buffer for all associations in our architecture. Each application entity must distinguish its MMS-PDU at the reception. At emission, the MMPM needs to know the destination application entity. Hence, the MMS-PDU must be encapsulated. This operation of encapsulation is necessary to route the MMS-PDU through the corresponding AAL connection.

> *Emission*: Each application entity must specify the value of its AssoRef and the called AEid before adding its MMS service parameters to the emission buffer. When the MMPM receives these AssoRef/AEid values as a header for an MMS-PDU, it looks in the MMS/ATM correspondence table for the AAL-SAP that corresponds to this AssoRef. Then it encapsulates this MMS-PDU into an AAL-PDU, and sends it to the called node through the network.
>
> *Reception*: At the called node, (works also for the calling node with a confirmation primitive), when the MMPM receives the MMS-PDU at an AAL-SAP, it looks for the corresponding AssoRef/AEid parameter in the MMS/ATM table. Then it sends the AssoRef/Aeid value with the result of the decode procedure of the received MMS-PDU to the reception buffer.

The MMS/ATM address mapping

An association (which is a connection in the application level) supports the communication between two MAEs. If an MAE wants to create an association with another MAE through a network, the calling entity must give the address of the called entity to the lower layers. The lower layers used in our communication architecture define only the ATM addresses that are specific to each AAL connection.

The MMS association is characterized by the AssoRef/AEid parameter. This parameter identifies a local Association Reference (AssoRef) and a destination application Entity identifier (AEid). The AAL connection is identified by an AAL Service Access Point (AAL-SAP) and a VCI/VPI value [11].

We suppose that each AAL-SAP corresponds to only one MMS association.

After opening an association, a table of correspondence between MMS associations and AAL connections (Figure 2) is updated. This table maps each AEid to an AAL-SAP and VCI/VPI.

The MMS management environment state machine (EMPM)

This function controls the right sequencing of service primitives and contains an MMS management environment entity to allow the establishment of an MMS environment. It provides only point-to-point communication primitives at the MMS level associations.

This state machine passes the parameters to the ATM Environment Management (AEM). These parameters are necessary to initialize an AAL connection that will be used by the EMPM state machine to initialize the MMS environment. When an ATM environment is setup correctly, the EMPM can use this ATM environment to send primitives to the MMS environment management service (like Initiate, Conclude, etc).

OSI Functionality in MMS/ATM

Considering our MMS-ATM architecture, one may be concerned, initially, with the loss of functionality incurred when the OSI protocol stack is by-passed. The MMS/ATM links directly the application layer and the ATM adapter layer (AAL), effectively by-passing the network, transport, presentation and session layers. Below, we will examine the consequences of this choice.

At the current state of networking technology, the probability of transmission errors is negligible. So we can by-pass the transport and session layers that take care of error-free transmission and recovery. Because of the absence of the transport and network layers, the flow control must be implemented by using an MMS parameter that imposes a maximum number of outstanding requests for services [12]. The ATM technology provides some internetworking functionalities of the network layer. It switches ATM cells from source to destination. Also, we have defined a table of correspondence between MMS associations and AAL connections, which achieves the function of address mapping, performed in a classic network layer.

After this analysis we can conclude that the loss of OSI functionality in our MMS/ATM communication system can be considered negligible with respect to the performance gain due to non-implementation of the network, transport, presentation and session layers.

IMPLEMENTATION

The purpose of this section is to give the reader an overview of the choices taken when implementing MMS/ATM. With this information it will be possible to understand the influence of these choices on the performance of the MMS/ATM stack. For this realization we have adapted a hybrid hardware/software implementation. This hardware/software system is composed of:

Software Part: Consists of an Ada program, which realizes MMS communication tasks. An Ada programming language is used to accelerate the software development process [12], because it has been designed for the implementation of real-time embedded systems.

Hardware Part: Consists of an ATM communication board that implements the first three layers of the MMS/ATM system. We have chosen the ATM Adaptation Layer type 5 (AAL5) because this lightweight protocol is sufficient for classical data and most types of real-time traffic. The AAL layer is implemented in hardware.

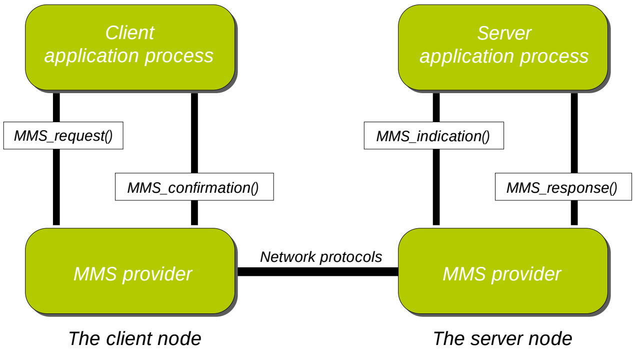

Figure 3. MMS Communication Sequences

In the following section, we discuss, first, the size of the PDU buffer. Then we give a method to manage the emission/reception buffer. Next, we present the ASN.1 encoding/decoding function used in our implementation and we describe our ATM communication board. Finally, we define the communication interface between the ATM board and the MMS application layer.

Size of the PDU Buffer

The choice of the PDU (Protocol Data Units) buffer size depends on the size of the messages to be transferred. In industrial environments the messages are relatively short. In the case of the Mini-MAP [4] architecture, the maximum length of PDUs is 1024 bytes. This restriction is not dictated by the MAC layer, which can transmit up to 8 Kbytes, but by performance considerations, namely bounds on token rotation times [13]. This problem does not apply in the case of our architecture since we use a high speed ATM transmission. The only limitation is the size of the segmentation and reassembly buffer managed by the AAL component. The length of this buffer is fixed to 128 Kbytes. For example, 4 Kbytes is the average message length when we are operating with 32 connections.

For our MMS/ATM implementation the size of the mono-block PDU buffer is 1024 bytes. This choice, motivated by who affirms that choice, allows a lower cost at encoding/decoding time [13].

Emission/Reception Buffer Management

Buffer management is a critical aspect in protocol implementation. It must be carefully designed if performance is an important factor of the development. There are two strategies for the choice of the architecture of an emission/reception buffer: mono-block or multi-block. The first approach is to allocate one buffer for all associations created by the MMS application entity. The disadvantage of this first approach is the wasting of a considerable amount of memory when we have a little number of associations. The second approach is to allocate small buffers for each association. This approach reduces the amount of unused memory but has the disadvantage of being more complex to manage. For reasons of simplicity, we have chosen to implement the first approach.

In this way it is necessary, in our opinion, to give a mechanism that must control the buffer emission and reception (Figure 3) The communication between two application entities making use of MMS is made on the basis of associations. When an MMS association is established, the maximum number of pending MMS requests is negotiated between the AEs. There are neither transport nor network layers in our MMS-ATM architecture. This is why we have chosen to implement the flow control by using a new MMS parameter that controls the padding of the emission and reception buffer.

This MMS parameter is specific to a client and a server application process. When an MMS client application sends a request service to an MMS server application, the client must indicate to server the state of his reception buffer (empty, overflow) by using a Buffc parameter. Inversely, an MMS server application must signal to an MMS client application the state of his reception buffer (empty, overflow) by using a Buffs parameter.

ASN.1 Encoders and Decoders

In Mini-MAP, a single transfer syntax may be used to encode and decode MMS PDUs. We will use the same syntax for the MMS-ATM architecture. The ASN.1 basic encoding rules [14] have numerous options. The main option concerns the way in which the length field of a composed type is encoded. There are tow possible formats: the definite format and the indefinite format. The definite format of encoding is much slower than the indefinite one [13]. This is why we have decided to encode in indefinite format.

There are, essentially, two approaches for parsing application PDUs (decoder) described in ASN.1. With the table-driven approach, the syntax of PDUs is reflected in tables which are used by a protocol independent of the parsing routine. On the other hand, in the code-driven approach, the syntax of PDUs is reflected in the control structure of the parser, which is protocol-dependent. We have chosen to work with code-driven parsers because of their speed.

ATM Interface Communication

The ATM board (Figure 4) offers a communication service interface for the workstation and incorporates a 4 by 4-switching element. The chosen physical interface operates at SDH STM-1 (155.22 Mbps). The switch and the header translator are the principal components of the communication system. The switch routes incoming cells to the workstation (or to the network) depending on the VCI and VPI values in the cellheader. The routing tables are set or altered dynamically by the signaling management software when connections are opened or closed.

The ATM Adaptation Layer type 5 (AAL5) chosen is hardware implemented. Communication between the AAL chip and the microprocessor is done by means of a shared memory and programmable interrupts (on a message or on a set of cells). A double port memory (DPRAM) has been chosen as the segmentation and the reassembly buffer. This avoids huge traffic on the board's 32-bits bus. If traditional memory (one access) is used, all data would have been carried twice on the bus, which would require high bandwidth on the board bus. Nevertheless, DPRAM modules have very large sizes and low capacities. In order to concurrently process several connections, it is necessary to have about one megabyte of capacity or more. Therefore the solution chosen consists of a small DPRAM (128 KB) as a temporary buffer. This ATM board is connected to the high speed PCI bus of the workstation.

Communication between ATM board and MMS application layer

The MMS Protocol Machine (MMPM) contains a set of application primitives to access the ATM driver (Figure 5). These primitives are of two types: signaling primitives and user primitives. The first type is used by the MMPM to execute signaling functions such as initialization or termination of a connection. The second type allows MMPM to send and receive data.

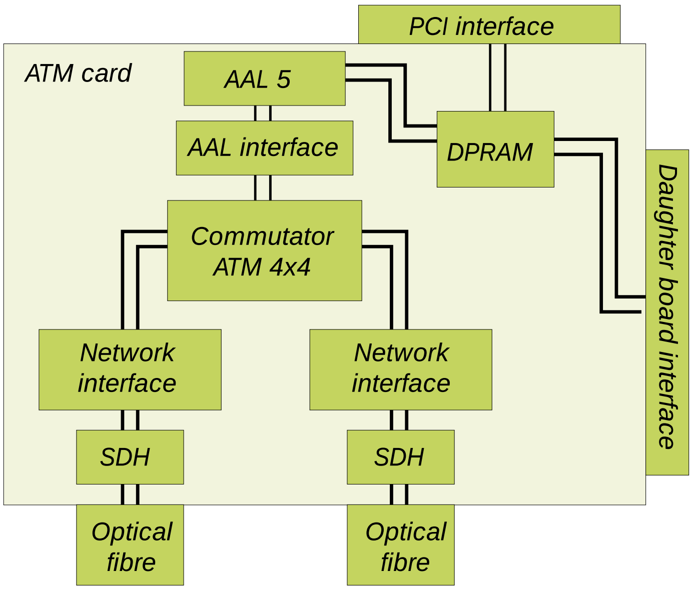

Figure 4. Communication Board.

The user primitives allow to:

Release an ATM port when the transfer is terminated (Close),

Receive data from an ATM port (Recv),

Send data through an ATM port (Send),

Create an ATM port attached to an access point (Attach).

The signaling primitives allow to:

Close a connection after the reception of an application request (end of treatment) or forced by the network (Kill),

Deactivate the ATM board (Close),

Initiate the ATM board (Init),

Open an ATM connection (Connect).

to be continued in Q2 . . .

Dr. Brahim Maaref received the degree diplome in Physics, Microelectronics from the University of Monastir (Tunisia), in 1993, the master's degree in Microelectronics from the University of Joseph Fourier, Grenoble (France), in 1995, and the Ph.D. degree in Microelectronics from the National Institute polytechnic of Grenoble (France) in 1999. His research interests include real-time communication systems for industrial automation.

Salem Nasri received his Doctor degree in Automatic Control and Computer Engineering from the National Institute of Applied Sciences of Toulouse, France in Juin 1985. Since 1988, he has been an assistant professor in Electrical Engineering in the National School of Engineering of Monastir, Tunisia. His research interest is in the field of Industrial Local Networks and Multimedia Applications. He is currently working in collaboration with the Logiciels, Systémes et Réseaux Laboratory.