

# Implementation and Performances Evaluation of Manufacturing Communication System Over ATM

continued from the previous issue . . .

## PERFORMANCE EVALUATION

The goal of the performance evaluation is to determine the performance improvement provided by the MMS-ATM architecture. The measurements are limited to the response times for MMS request-responses between two MMS-ATM nodes. We have used a part of an Ada implementation of the Mini-MAP architecture [15]. The MMS standard is composed of 86 services. Our purpose is not to implement all the MMS services but to design a communication system that will provide an efficient communication in an MMS environment. Our implementation provides only the execution of a subset of MMS services defined by the standard. These services are Initiate, Start, Write, Read, Download, and Conclude.

### Platform

An experimental platform implementing our communication system has been developed in our laboratory to evaluate the performance of MMS services in real operational conditions. It is composed of:

- **A Workstation Client:** Personal Computer (PC), Pentium 133 MHz, 32MB RAM, WindowsNT/DOS, ATM board.
- **An Operator Station Server:** Personal Computer (PC), Pentium 200 MHz, 32MB RAM, WindowsNT/DOS, ATM board.

A GNAT (Ada95) compiler is used to develop and evaluate the performance of software components.

### MMS services description

In the following, we describe the MMS services that are implemented in our system.

- **Read service:** the client uses the read service in order to request that a server returns the values of one or more variables.
- **Write service:** the client writes variables at the server. In the service call a specification of the variables with their corresponding values is transmitted.
- **Initiate service:** The Initiate service is used to establish a communication between two MMS-users. The Initiate

service negotiates the conditions for a communication with the MMS-partner.

- **Conclude service:** The Conclude service is used to terminate a communication in a defined order.
- **Start service:** This service starts the Program Invocation. The Program Invocation must be in the Idle State for this service to be valid.
- **Stop service:** The Stop service causes a transition from the RUNNING to the STOPPED state of the Program Invocation. The progress of the Program Invocation is then stopped.
- **Download service:** The Download service is composed of three services. The first service is the Initiate Download service. It is used by the client to request the server to begin loading. The second service is the Download Segment service. It is used by the server to ask the client to start downloading after receiving the Initiate Download service. Finally, after the server has received all the segments, a Terminate Download service is sent to the client.

In the case of a confirmed service, the measured time T represents the duration between the transmission of a request and the reception of the confirmation by the client program (Figure 6). In the case of an uncon-

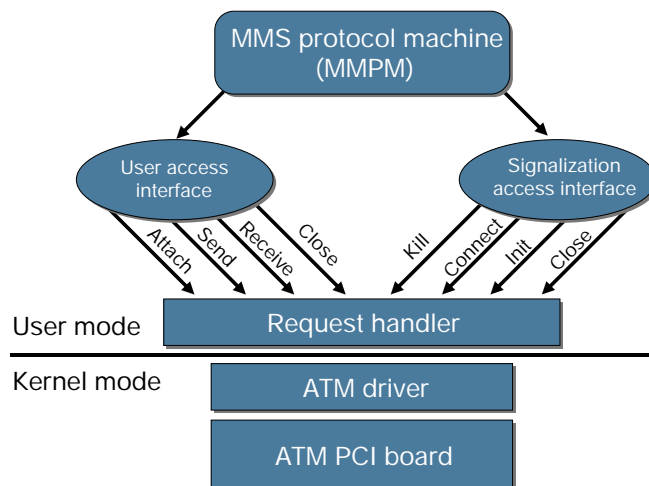


Figure 5. ATM Board Interface.

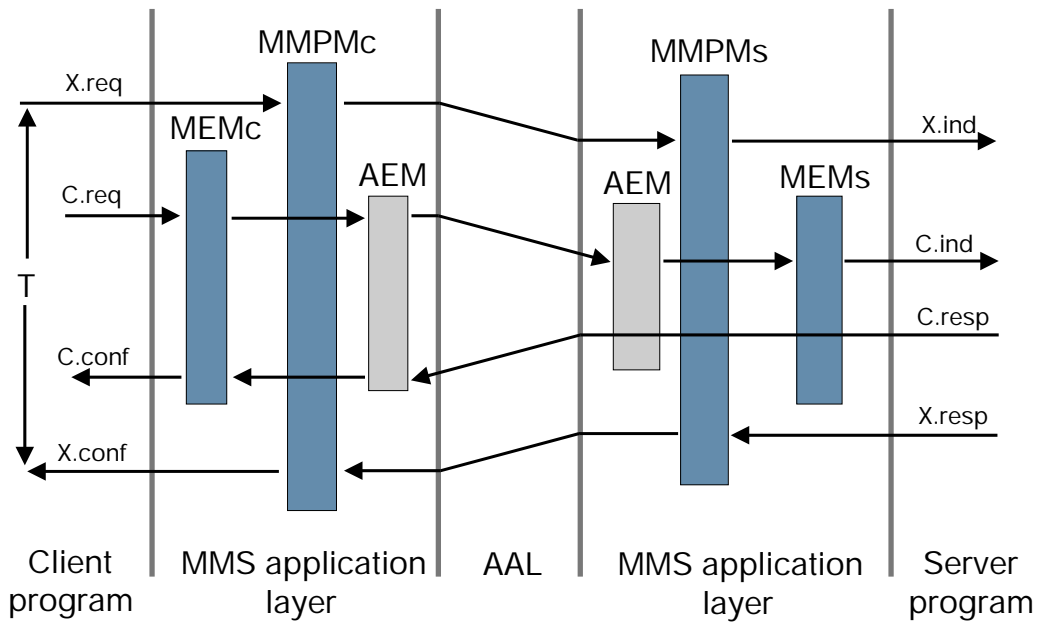


Figure 6. Transmission of an MMS request and its response.

firmed service, the transmission delay is the time between the transmission of a request by the client program and the reception of the indication primitive by the server program.

Some MMS services are needed to manage the ATM environment. Primitives C in Figure 6 represent these MMS services which are:

- **Link:** This service is used by an MAE to establish an MMS environment. This MAE must send to the MEM all parameters necessary to initiate an MMS association and ATM connection.
- **ATMinitt:** This service defines the ATM signaling protocol parameters necessary to establish an AAL connection. It is used to initiate an ATM environment.
- **ATMclose:** This service is used by a MEM service to close an AAL connection.

### Measurements

Measurements have been classified into two categories:

1. External measurements: these measurements consider the implementation as a black box and provide response times for MMS request-responses between two MMS/ATM nodes.
2. Internal measurements: the implementation is evaluated from the inside to identify the time spent in different parts of the software. This includes PDU encoding/decoding, the MMPM state machine, ATM transfer, and the VMD program.

#### External Measurements

In this subsection, we focus on the performance perceived by user applications communicating through MMS. The measurements were evaluated by an MMS application program. The time for each service given in Table 1 is determined by sending a large number of

these services, recording the time for the total transmission and computing the average time per service request. The variables used in the server for experiments with the Read and Write services are of type integer with 2 bytes for name length and 4 bytes for value length. Table 1 gives the request-response times of the Read, Write, Initiate, Conclude, Start, Stop, and Download services.

We note that as the number of variables read or written increases, the response time increases. This cannot be explained by the existence of the AAL or ATM layers. Indeed, the reason for this increase is that the response time depends on the application layer and particularly on the encoding/decoding process.

We have also measured the response time of the Download MMS service for two segment sizes: 512, 1024 bytes. Table 4 indicates the Download time as a function of the segment number of 512 and 1024 bytes.

#### Internal Measurements

In the previous paragraphs, we have considered the implementation of MMS-ATM as a black box that provides response times for MMS request-responses. This

MMS Service	Time (ms)
Read	4.30
Write	4.80
Initiate	4.70
Conclude	4.00
Start	4.30
Stop	4.10
Download	10.10

Table 1. Average Times for some MMS services.

Services	Read		Write		Download	
Component	Time (ms)	Percent	Time (ms)	Percent	Time (ms)	Percent
VMD	0.50	11.6	0.60	12.5	0.60	5.9
MMPM	1.60	37.2	1.60	33.5	4.00	39.6
Encoder	0.80	18.6	1.00	21.0	2.00	19.8
Decoder	0.60	14	0.80	16.5	1.50	14.9
ATM Transfer	0.80	18.6	0.80	16.5	2.00	19.8
Total	4.30	100%	4.80	100%	10.10	100%

Table 2. Execution Time for Read, Write and Download services.

Number of variables	Read Service (ms)	Write Service (ms)
1	4.30	4.80
10	4.40	4.85
100	4.60	5.05
150	4.70	5.20
200	4.80	5.30
250	4.90	5.40

Table 3. Average Times for Read and Write services.

response time is determined by the execution time of five main components: MMPM, Encoder, Decoder, API and ATM transfer (Table 2).

Table 3 shows in detail the execution time of Read and Write services of one variable (integer that has 2 bytes for the name length and 4 bytes for the value length).

We have also measured the execution time of the MMS Domain service, which allows the transfer of data presented as segments. The response time for request-responses of the domain download service depends on the segment size. For a segment size of 512 bytes, the response time is 10.1 ms. Detailed values of these measurements are given in Table 4.

## ANALYSIS AND COMPARISON

### Analysis

Our results of external measurements show that the latency for the Download service is nearly twice as large as those for other services. This can not only be explained by the difference between encoding/decod-

ing time of Download service and other services. The explanation is that the Download service involves twelve transfers through all the communication layers at the source and destination station. However, for the other services there are only four transfers through all communication layers at the source and destination station.

Our implementation allows the execution of the Download service with segment sizes greater than 1024 bytes. For these segment sizes, the response time increases significantly. This is due to the number of transfers performed between the emission/reception buffer and the reassembly/segmentation buffer of our ATM communication board. For example, a 512 bytes segment requires two transfers and a 1024 bytes segment requires three transfers.

The internal measurements show that the MMPM execution time is greater than that of the encode/decode together and is twice as long as the ATM transfer time. This can be explained by the fact that MMPM connects seamlessly the MMS and AAL layers. MMPM runs some functions necessary for this connection

Number of segments of 512 bytes	Time(ms)
1	10.1
2	13.2
3	16.9
4	20.4
Number of segments of 1024 bytes	Time(ms)
1	12.5
2	16.0
3	21.6
4	25.7

Table 4. Average Times for Download service.

such as: encapsulation, address mapping, and transfer. Details on the address mapping and encapsulation functions are provided in the sections 4.3 and 4.4 respectively. The transfer function includes all low level transfer routines from the host to ATM-card.

## Comparison between MMS/ATM and other architectures

Now, we try to compare the performance of our MMS/ATM communication system with the Mini-MAP and MAP architectures.

The Mini-MAP architecture that is being considered consists of three layers: MMS application, data-link, and physical. The data-link layer is subdivided into the MAC (Medium Access Control) and LLC (Logical Link Control) sub-layers. The MAC sublayer conforms to the IEEE 802.4 token passing bus standard (physical layer). The LLC sublayer offers two services: an unacknowledged connectionless service (LLC type 1) and an acknowledged connectionless service (LLC type 3). The Mini-MAP application layer offers the services of a reduced subset of OSI application service elements, namely MMS, network management and a directory called the object dictionary.

A major issue in manufacturing systems is the need for rapid and reliable communication among the wide variety of computers and automated equipment involved in the manufacturing process. Typically, the factory floor devices are incompatible with one another. A standardized communication network is needed to allow their interconnection.

General Motors started the specification of a network standard for factory communication, called MAP. MAP relies essentially on a standard connection oriented ISO profile. It implements all of the ISO protocol layers. Note that Mini-MAP MMS does not differ from MAP MMS in the syntax of messages but only in the underlying support called MMPM that governs the message exchanges between the MAEs [13].

We have compared our MMS/ATM implementation with few existing MMS implementations. In most cases [16], our implementation has achieved a better performance. In the left cases, the performance is equivalent to them. But even for these cases, our implementation is advantageous. This is due to:

- The use of ATM technology for a communication network has some technical advantages. In classical MMS/MAP (or MMS/Mini-MAP) implementations the number of interconnected workstations are bounded, as well as that of logical segments. In general, to interconnect several machines from different segments the designer may use bridges or gateways. This increases the network equipment and management cost. In contrast, there are no such constraints in our MMS/ATM implementation. Furthermore, it is worth to use ATM, since its virtual connection concept introduces flexibility in the management of the network connections.
- ATM networking provides end-to-end quality of service (QoS) guarantees [17]. It offers:
  - Transparent handling of different data types,
  - High requirements for multimedia applications

(latency, bandwidth),

- Multicast support and
- Real-Time support.

Some modern applications of multimedia communication derived from the auto industry are an example of the use of this technology in design, manufacturing and sales [18]. Our MMS-ATM architecture offers high bandwidth (155 Mbps) that allows integration of innovative applications as multimedia in industrial environment.

Some examples of multimedia applications in manufacturing systems are: Tele-surveillance and the journal management. The Tele-surveillance application in a manufacturing environment provides a means for the control of remote sites. Remote cameras and microphones are used to capture image and audio data that are sent to displays, loudspeakers, storage systems, or to specific signal processing systems. A Tele-surveillance application defined here is similar to a robotics application. The journal management application provides means for recording time-stamped image and audio information related to specific events

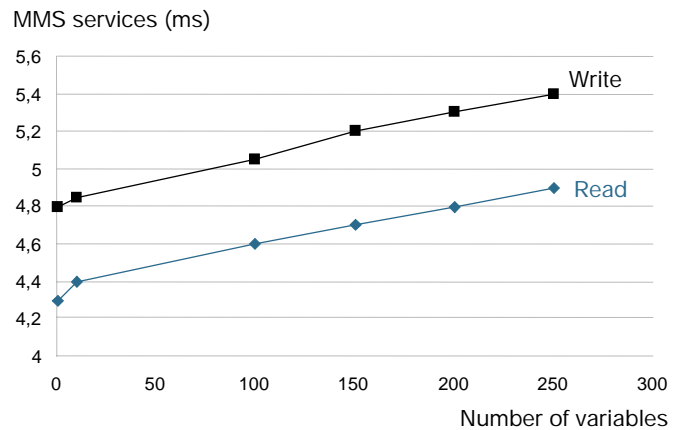


Diagram 1. Average Times for Read and Write services.

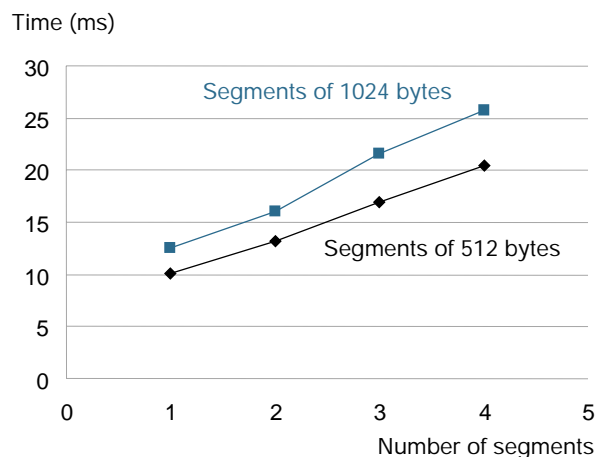


Diagram 2. Average Times for Download service.

***AD VMIC***

that are likely to occur on a manufacturing plant.

These applications can be used to measure the positions of parts and guide the assembly process to reduce the need for costly and inflexible fixturing. It can also be used to monitor and maintain quality [19].

## POSSIBLE IMPROVEMENTS

### Introduction

The use of both software and hardware in communication system implementation seems to be a key issue in high performance networking [20]. Some parts of a communication system are implemented in hardware to achieve better performance, while other parts are preferably implemented in software, thereby yielding greater flexibility.

The goal of hardware/software design is to produce an efficient implementation that satisfies the performance and minimizes the cost, starting from the initial specification. However, there exist a diversity of technological solutions based on available hardware/software components. The designer needs to explore the possible solutions either using automated tools or selecting manually his choices. Different solutions are analyzed in order to choose the best solution from trade-offs between both technologies. The latter result is the one that satisfies more the required performance. At any stage of the design, the user can cancel one or several design steps in order to explore new choices.

The MMS/ATM communication system presented in this paper is composed of two parts: hardware and software. We have evaluated and analyzed in section 5.3 the performance of such a system. As a result, we have noted that the transmission delay of MMS services depends on the performance of the software part. Our objective is to improve the transmission delay of MMS services by implementing the software part which contains some crucial tasks such as encoding and decoding process.

### Transmission delay

In this subsection, we study the transmission delay of MMS services. We are interested in the delay caused by the execution of protocol layers, in particular, the execution of MMS/ATM communication sublayer. We calculate the percentage of the execution time in this sublayer with respect to the total time of the MMS service transmission delay. This percentage is calculated for different robot response times.

The transmission delay of the MMS Write service is composed of seven times (Figure 1).  $t_1$  and  $t_6$  are the execution times of client and server application entities, respectively.  $t_3$  and  $t_4$  represent the transmission

times between two MMS/ATM nodes. The execution times of the MMS/ATM communication sublayer at client and server node are, respectively,  $t_2$  and  $t_5$ . Finally,  $t_7$  is the response time of an industrial equipment (e.g. a robot).

We have defined three parameters:

- execution time ( $t_e = 2*(t_2 + t_5)$ ) of the MMS/ATM communication system software part (MMS/ATM communication sublayer). The time  $t_e$  correspond to execution time of MPPM, Encoder, and Decoder modules of Write service (see Table 2).
- global execution time ( $t_c = 2*(t_1 + t_2 + t_3 + t_4 + t_5 + t_6) + t_7$ ) of the MMS Write service. The time  $t_c$  is equal to MMS Write execution (see Table 1) plus the response time of an industrial equipment.
- and the ratio :  $R = t_e/t_c$ .

When the response time of a robot is equal to 100 ms, the execution time ( $t_e$ ) of the MMS/ATM communication sublayer, which is a software part, is not significant (3.2% $\leq$ R $\leq$ 13.7%). But this execution time ( $t_e$ ) becomes significant (23% $\leq$ R $\leq$ 58%) when the response time of a robot is less than 10 ms (Table 5). In this case, a hardware/software implementation of the MMS/ATM communication sublayer is recommended to satisfy the real-time requirements of industrial applications.

By implementing an MMS/ATM rather than an MMS/TCP/IP/ATM, we think that we have implemented a faster communication system. However, Table 5 indicates that the overhead incurred due the MMS/ATM communication sublayer, which is a software implementation, can reach 58% of the overall communication time. This justifies a hardware/software implementation of this sublayer, hoping to achieve better performance.

### 7.3- Design Space Exploration of the MMS Protocol

The objective of the exploration of the design space presented in this work is to analyze the significant hardware/software implementations and to illustrate the advantages of mixed hardware/software integration for a communication protocol. The analysis of several solutions is achieved manually, but the performance estimation of each module is done using different simulation and synthesis tools. Each module is described, synthesized, and simulated in software and in hardware to determine execution times.

#### Target Architecture

An example of the target architecture onto which a mixed hardware/Software system will be mapped, is shown in Figure 7. This architecture is based on a PC

$T_r(ms)$	100	50	20	10	5	2	1
$T_e(ms)$	3.4	3.4	3.4	3.4	3.4	3.4	3.4
$T_c(ms)$	104.854.8	24.8	14.8	9.8	6.8	5.8	
$R = t_e/t_c(\%)$	3.2	6.2	13.7	23	34.7	50	58

Table 5. The Ratio R of Write Service.

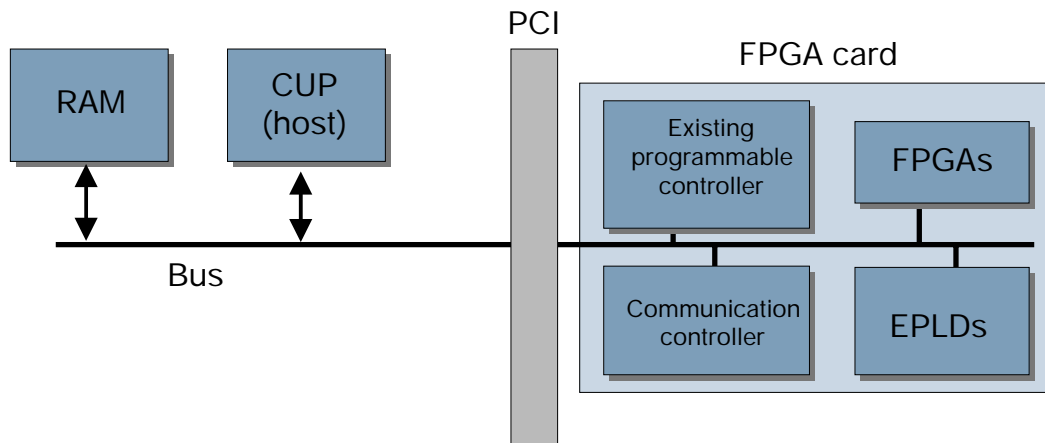


Figure 7. Target Architecture.

(Personal Computer) with an extension board (FPGA).

The platform includes several standard peripheral components such as I/O ports (serial, parallel), Direct Memory Access (DMA), Interrupt controllers, etc. Their drivers may be kept in EPROM or may be downloaded from an external memory.

In our approach, a modular, flexible and distributed architecture is used. This architecture serves as a platform over which a mixed hardware/software system is mapped. The architecture is composed of three kinds of components: software, hardware, and communication. A software sub-system consists of a sub-set of tasks running on a microprocessor (host CPU). A hardware sub-system consists of custom synthesized hardware modules (FPGA card). A communication controller may be implemented as a hardware or a software sub-system. This controller implements a communication protocol between software and hardware tasks.

#### *Hardware and Software Modules Performance of the MMS Protocol*

For the sake of simplicity, we will restrict the types of components used. A software component will be executed on the host processor and a hardware component will be represented by a Xilinx 4010 FPGA (Field Programmable Gate Array).

We have chosen to implement the encoding/decoding process in hardware for the following reasons:

- It is independent of the communication stack.
- It presents an average of 33% of the transmission delay of MMS services, which is not negligible.

The other tasks of the MMS/ATM communication sub-layer depend on the lower layers (AAL layer in our case). To preserve the flexibility of our implementation, these tasks remain as a software program.

The execution times are obtained by the simulation of critical paths of encoding/decoding modules. For hardware, the execution time is deduced using results of simulation with a VHDL description. The frequency of the Xilinx 4010 FPGA component is equal to 10 Mhz.

In the Table 6, we give the new average times of MMS request-response services. We obtain an average gain

of 25% percent with respect to the values given in table 1. In order to analyze different hardware/software implementations, the cost of each component is needed. There is no exact measure to evaluate the hardware and the software implementations. However, the hardware implementation is more expensive than a software implementation. But a hardware/software solution meets high-performance with a reduced design cost.

## CONCLUSION

In this paper, we have studied the implementation structure and the resulting performance of an MMS/ATM communication system. Then we evaluated the performance of this implementation from an external and an internal point of view.

The external measurements show that the response times of MMS services are high for some time-critical applications. We have expected that the use of ATM technology (high speed) and the minimization of the number of protocol layers will yield very small response times. We have carried out internal measurements in order to understand the partition of this

MMS Service	Time (ms)
Read	3.30
Write	3.15
Initiate	3.15
Conclude	3.10
Start	3.45
Stop	3.20
Download	6.95

Table 6. A new Average Times for some MMS services.

transmission delay in each part of our MMS/ATM communication system and to propose a solution to lower this delay.

The internal measurements have shown that approxi-

mately 80% of the MMS request-response time are spent in the application layer of the MMS/ATM stack. Then, we have studied a hardware/software implementation of the MMS protocol. This kind of implementation allows improving the transmission delay of MMS services with an optimal cost. As a result, the time spent in the application layer becomes approximately 55% per cent of the MMS request-response time.

Experience made with an Ada run-time system has shown that memory management is a key factor for the performance of the encoding and decoding process. We have chosen to implement the encoding/decoding process in hardware on a co-processor. This hardware implementation allows a better encoding/decoding performance avoiding any memory management problems ■

*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, Systèmes et Réseaux Laboratory.*

## REFERENCES

1. ISO/IEC. 9506-1, 1990, Industrial Automation Systems - Manufacturing Message Specification, Part 1: Service definition. International Standards Organization.
2. ISO/IEC. 9506-2, 1990, Industrial Automation Systems - Manufacturing Message Specification, Part 2: Protocol Definition. International Standards Organization.
3. M. DE PRYCKER, 1991, Asynchronous Transfer Mode, Ellis Horwood.
4. G. Motors, 1988, Manufacturing Automation Protocol, Version 3.0.
5. W. RANDY HEUSER, Wm RANDY, CHEN, RICHARD, and STOCKETT, MICHAEL H., 1993, Using Manufacturing Message Specification for Monitor and Control at Venus, Proceedings of the Network Technology Conference, NASA Conference Publication 3241.
6. IEC 870-6-503, 1996, TASE.2 Services and Protocol, ICCP, Inter-Control Center Communications Protocol, Version 6.1.
7. M. DECINA and V. TRECordi, 1992, Traffic Management and Congestion Control for ATM Networks, IEEE Network, September 1992.
8. A. ECKBERG, B-ISDN/ATM Traffic and Congestion Control, IEEE Network Magazine.
9. Q. ZHENG, K. G. SHIN, and C. SHEN, 1993, Real-Time Communication in ATM Networks, Proceedings of the Real-time Systems Symposium.
10. A. VALENZANO and L. CIMINIERA, 1990, Performance Evaluation of Mini-MAP Networks, IEEE Transaction On Industrial Electronics, 37(3), pp. 253-258,
11. ITU-T recommendation I.363 - B-ISDN ATM Adaptation Layer (AAL) specification - March 1993.
12. J. G. P. BARNES, Programming in Ada, Addison-Wesley, 1989.
13. M. Li, P. PLEINEVAUX and F. VAMPARYS, 1994, Performance Evaluation of a Reduced OSI Stack,

ATM :	Asynchronous Transfer Mode	OSI:	Open System Interconnection
AEM :	ATM Environment Management	PDU:	Protocol Data Unit
AAL:	ATM Adapter Layer	PVC:	Permanent Virtual Connection
CS:	Convergence Sublayer	QoS :	Quality of Service
EMPM:	Environment Management Protocol Machine	SAR:	Segmentation And Reassembly sublayer
IUT-T:	International Telecommunication Union (ITU-T) and Telecommunication	SAP:	Service Access Point
LLC:	Logical Link Control	SVC :	Switched Virtual Connection
MMS:	Manufacturing Message Specification	UII :	Use-User Information
MAE :	MMS Application Entity	UNI:	User Network Interface
MEM :	MMS Environment Management	VMD:	Virtual Manufacturing Device
MAP:	Manufacturing Automation Protocol	VCI :	Virtual Channel Identifier
MMPM:	MMS Protocol Machine	VPI :	Virtual Path Identifier
MAC :	Medium Access Control	VC :	Virtual Connection

Appendix

- Internetworking: Research and Experience, 5, pp. 71-87.
14. ISO 8825, Information Processing Systems, Open Systems Interconnection, ASN.1 Basic Encoding Rules.
  15. H. NUSSBAUMER and P. PLEINEVAUX, 1996, Ada implementation of Mini-MAP architecture, Swiss Federal Institute of Technology, Lausanne, Industrial Computer Engineering Lab.
  16. P. CASTORI, P. PLEINEVAUX, 1995, A Generic Architecture for MMS Servers, Proceedings of the IEEE International Workshop on Factory Communication Systems (WFCS), Leysin, Switzerland, pp. 211-218.
  17. C. SHEN, 1996, On ATM Support for Distributed Real-Time Applications, Proceedings of the IEEE Real-Time Technology and Applications Symposium.
  18. CHWAN-HWA "JOHN" WU, M. C. YUANG and J. D. IRWIN, 1998, Multimedia and Multimedia Communications: A Tutorial, IEEE Transaction On Industrial Electronics, 45(1).
  19. S. ROSENTHAL, 1994, Vision Guided Assembly, em Industrial Computing, 13(6).
  20. S. FISCHER, J. WYTREBOWICZ, S. BUDKOWSKI, 1996, Hardware/Software Co-design of Communication Protocols, Proceedings of EUROMICRO.

FRONT END MODULE

user interface

World Computers Ltd. ( U.K. ) for world class embedded software components .  
Offices in London and Brussels .

**Save money and save time by using the Front End Module!**  
(also known as a Man Machine Interface Human Interface or User Interface)

- Full featured.
- Simple and easy to use.
- Provides all the features required for most front ends.
- Handles all displays, inputs, cursor and editing operations.
- Separates out screen design into compact binary coded scripts, display rendering into the display driver and display and input operations into the Front End Module proper.
- Low cost and fully supported.
- Saves typically 20 000 Euros or U.S. \$ in development costs and 3 months in development time.
- Allows front ends to be developed in hours to days .
- Compact code. Written in ANSI C.
- Comes with MSVC++ build for evaluation and prototyping .

Check our site for full information:  
[www.wrldcomp.com](http://www.wrldcomp.com)