MMS Beyond MAP—An Integration Protocol for Factory Networks

Mel Hagar, Texas Instruments Corporation
Terry Bass, Digital Equipment Corporation
Carl Raff, Savoir

National Center for Manufacturing Sciences
November 1990

Coordination
Chris Conrad
Bresson Rupp Lipa & Co.

Editorial Functions
Jeff Holtzman
Bresson Rupp Lipa & Co.
April 22, 1991

© Copyright 1990
National Center for Manufacturing Sciences
All Rights Reserved
Printed in the United States of America

National Center for Manufacturing Sciences
900 Victors Way
Ann Arbor, MI 48108
(313) 995-0300

Table of Contents

Abstract 4
The Problem 5
What is MMS? 7
History 7
MMS and the OSI Stack 9
Inside MMS 13
Client-server Model 13
MMS Objects 13
MMS Services 14
Extensions and Companion Standards 15
Section Summary 16
MMS in the Big Picture 17
Vendor Support 17
Interoperability 17
Software Portability 18
Migration 18
MMS Versions 19
MAP and MMS 19
Conformance Testing 19
Benefits and Risks of MMS 20
Benefits 20
Risks 20
Conclusions 22
References 23

List of Figures

1. MMS and the OSI Stack 6
2. Historical Development 8
3. The OSI Stack 9
4. OSI Packaging 10
5. Low-level OSI Access Schemes 12
6. MMS Network Architectures 12
7. MMS Interface 13
8. Virtual Manufacturing Device Model 14

List of Tables

1. OSI Stack 10
2. VMD Objects 14
3. MMS Services 15
4. Companion Standards 15

Abstract

MMS (Manufacturing Message Specification) is an International Standards Organization (ISO) standard OSI application layer protocol that aids communication among disparate programmable manufacturing devices. This paper provides an historical and technical overview of MMS, and also discusses its relations to underlying network technologies. The paper then goes on to discuss how MMS fits into the larger scheme of things. The final section discusses risks and benefits of using MMS. An appendix provides references for more detailed information.

The Problem

In today's competitive global economy, manufacturing must become more efficient and better integrated with the rest of the enterprise. However, attaining greater efficiency and integration requires significant technical evolution. Work done by major corporations and standards organizations during the past decade has gone a long way toward defining the required technology.

The goal of this technology is simple: improved communication, both within manufacturing and among the divisions of the enterprise as a whole.

To increase manufacturing efficiency, machines on the factory floor must learn how to communicate with one another. To increase integration with the rest of the enterprise, and thereby increase overall efficiency and competitiveness, manufacturing machinery must also learn how to communicate with the business and technical machines in the front office.

With these types of communications channels in place, the overall manufacturing process will become more efficient, because each stage in the process can be dovetailed to the preceding and succeeding stages.

In addition, better communication with other divisions of the enterprise can provide better inventory control, faster response to changing market conditions, better coordination with both suppliers and customers, and other benefits.

However, achieving this type of communication has proven to be frustratingly difficult, and most of the difficulty arises from a lack of communications standards. In the early years of manufacturing automation, there was little reason for devices from different vendors to be able to communicate with one another. As time passed, methods of communication multiplied into a veritable Tower of Babel, producing the well-known "Islands of Automation" that represent only partial solutions to the problem of achieving greater efficiency and integration.

The good news is that, due to the strenuous and sometimes painful standards work done during the 1980s, achieving truly computer integrated manufacturing (CIM) is now more than just a gleam in the eyes of some dreamers. In fact, as we enter the 1990s, many pieces of the CIM communications puzzle have fallen into place, and others are on the verge of doing so.

The first piece of the puzzle to fall in place was the definition of the seven-layer Open Systems Interconnect (OSI) reference model, best known through independent recommendations in the U.S. Government OSI Profile (GOSIP), in the Technical and Office Protocol (TOP), and in the Manufacturing Automation Protocol (MAP).

Early efforts in the definition of MAP focused on the lowest levels of the stack: cabling and access control. As compromises on those issues were reached, focus shifted to higher and higher levels of the stack, and thus was born the Manufacturing Message Specification (MMS).

MMS is a communications protocol that aids the process of integrating a heterogeneous factory environment consisting of programmable sensors, robots, controllers, numerical-control (NC) machines, etc.

MMS was designed to operate on a network based on the seven-layer OSI stack; as shown in Figure 1, MMS resides in layer seven of the OSI stack.

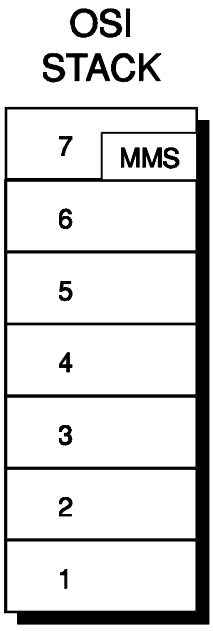

Figure 1: MMS and the OSI Stack

Because of the way MMS evolved, there is public misperception about what it can and cannot do, and about its relationship to the underlying network technology. To help correct that misperception, this paper will

* Provide a technical overview of MMS
* Discuss the origin and evolution of MMS
* Discuss benefits and risks of using MMS

What is MMS?

In this section, first we will discuss the evolution of MMS and MAP. Next we will discuss the OSI model and show where MMS fits into it. Then we will provide a technical introduction to MMS and the services it provides. Last we will discuss extensions to MMS.

History

As factory automation grew during the 1960s and 1970s, it became increasingly apparent that simply automating isolated machines was not enough; somehow they also had to be tied together, made to interoperate. With those requirements in mind, General Motors formed a group, the Manufacturing Automation Protocol (MAP) Task Force, in 1980.

The group's charter was to identify a means for inter-device communication on the factory floor. The group spent the next seven years defining and refining a set of recommended standards for a complete seven-layer OSI environment appropriate for operating a manufacturing plant. Several sets of recommendations were published, culminating in MAP 3.0, which was published in 1988. The MMS standard is specified by MAP. All versions of MAP are based on the seven-layer OSI model.

The problem is that the OSI model was designed for connecting general-purpose computers; further work was needed to meet the special needs of manufacturing devices. To meet those needs, the MAP Programmable Devices (PD) committee developed a protocol called the Standard Message Format (SMF) by 1984. SMF was the precursor of MMS.

General Motors released the first MAP specification (1.0) in April of 1984, and staged a demonstration of it, including SMF, at the National Computer Conference that summer. Although successful in generating industry interest, the demo showed that MAP (including SMF) still had a long way to go.

Later that year the PD committee combined efforts with an EIA group (WG1393) that was working on a similar protocol for CNC machines; the combined effort led to publication of the Manufacturing Message Format Standard (MMFS) in December of 1984.

"Memphis," as it was called, included most of the services required to write substantial factory-floor applications, but used an encoding method that was incompatible with other application-layer standards. In addition, the MMFS specification was vague, so there was significant possibility of incompatibility between different MMFS implementations.

To overcome those problems, a joint session of the GM and EIA groups decided in early 1985 to use Abstract Syntax Notation 1 (ASN.1). ASN.1 was developed by the International Standards Organization (ISO); it specifies a standard means of notating and encoding information in digital format. That decision removed the major remaining hurdle to the development of the MMS specification. So by mid-1985, the new Manufacturing Message Specification, or MMS, was in draft form.

Meanwhile, during 1985 and 1986, GM published several 2.x versions of MAP (which included MMFS). Eventually GM published an implementor's release of the MAP 3.0 specification in the spring of 1987; that specification included an early version of MMS.

However, although they were developed from a common model, there were incompatibilities among the MAP 2.X implementations, and, unfortunately, in resolving the 2.x compatibility issues, MAP 3.0 was incompatible with the previous versions. In addition, the MAP 3.0 specification was released before products were available, so customers were left in the unenviable position of choosing a known dead-end technology or waiting for a promised technology with an unknown delivery date. Further, MAP 3.0 required the lowest levels of the OSI stack to be implemented in ways that were disagreeable to customers with sizable investments in other technologies. The net effect was that credibility was lost, both for MAP and for MMS, even though MMS had already been weaned from MAP.

In retrospect, it is easy to see the difficulties in the development and release of the various versions of MAP. However, the MAP effort produced many positive results; many view MMS as the jewel of the MAP effort. In addition, it should be stressed that MAP was the first public implementation of the OSI model in the U.S., so it thereby forced consideration of many important issues, including international communications standards, testing, and certification. So although MAP itself has not achieved widespread adoption, its development has been highly beneficial for all industries concerned with communications standards.

Returning to MMS, the final version of the specification was approved in December 1988. Under agreement with the EIA, it was simultaneously released as EIA 511 and published in early 1989. The ISO published its version (including many errata) in October, 1990. EIA-511, together with its errata, is identical to ISO 9506, not counting typography. It is intended that EIA 511, ANSI Standard 19506, and ISO 9506 maintain compatibility henceforth. Figure 2 summarizes the historical development of MAP, SMF, MMFS, and MMS.

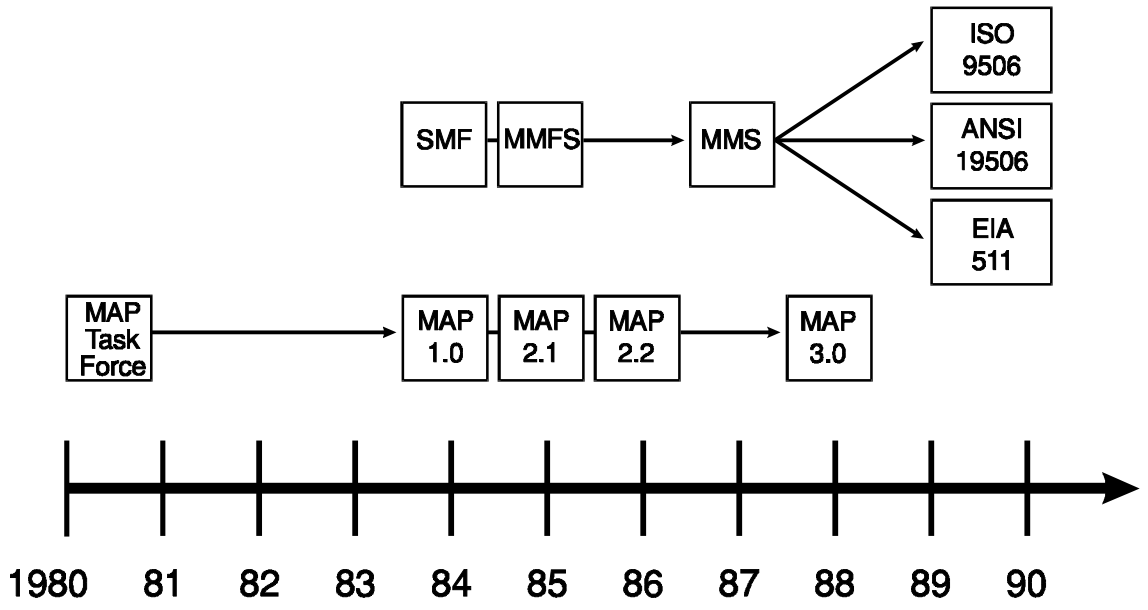

Figure 2: Historical Development

MMS and the OSI Stack

Throughout the 1970s and early 1980s, incompatible and proprietary communications protocols proliferated. In response, the ISO developed the OSI reference model, which defines a seven-layer stack. The OSI stack provides a methodology for diverse kinds of equipment from diverse vendors to communicate.

The OSI stack is a model, not an implementation; in actuality, there are many implementations. MAP was one of the first fully defined implementations; in the U.S., GOSIP is the most well known.

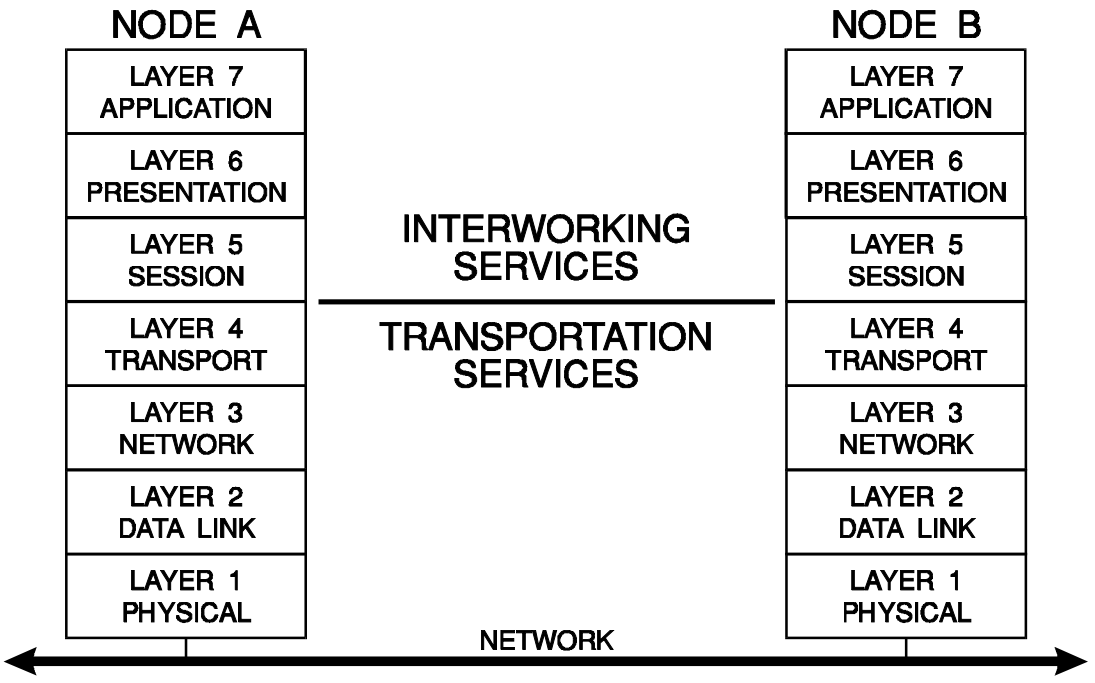

Figure 3: The OSI Stack

As shown in Figure 3, the OSI stack consists of seven layers. Those layers may be categorized in two groups, transportation and interworking. Transportation is the responsibility of the lower four layers; those layers provide reliable transfer of messages across a network. Interworking is the responsibility of the upper three layers; those layers allow cooperating applications to carry on meaningful conversations. The transportation group is more concerned with the physical transfer of information; the interworking group is more concerned with the meaning and structure of that information.

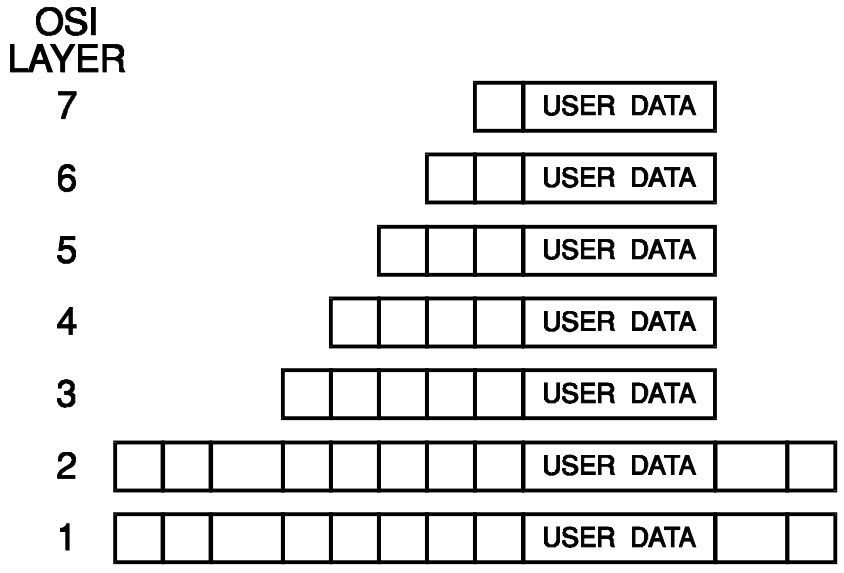

Figure 4: OSI Packaging

When two devices are communicating via an OSI stack, each layer logically communicates with the corresponding layer on the other device. Physically, data from an application program starts at the top of the OSI stack and travels down the stack. As it does, it is enveloped in progressively more information, as shown in Figure 4. The additional information routes the application data, ensures that it is transmitted error-free, and presents it in a consistent format on both devices. Table 1 summarizes the function of each layer.

Table 1—OSI Stack

Layer	Name	Description
7	Application	Includes the application program and services it uses to access lower layers. May include high-level functions such as file transfer (FTAM), directory services, message handling (MHS). MMS is one group of high-level services.
6	Presentation	Provides a consistent representation of information to two communicating devices. Overcomes differences in CPU architecture, operating system, etc.
5	Session	Controls data exchange after connection has been established by lower layers.
4	Transport	Provides reliable end-to-end data transfer across the network. May provide error correction.
3	Network	Provides routing information from source to destination across the network. May provide error correction.
2	Data Link	Assembles data into packets, provides addressing information, checks data and physical medium for errors. Often divided into two sublayers: Logical Link Control (LLC) and Media Access Control (MAC).
1	Physical	Data-encoding technique; does not include cables, connectors, modems, or other physical media.

The basic idea behind the layered approach is that any two layers separated by at least one layer need know nothing about how physical communications are carried out by the intervening layer or layers.

That is an important point because it shows why MMS is truly independent of MAP. As long as a given MMS implementation can talk to the associated layer-six implementation (which can in turn talk to the corresponding layer-five implementation, and so on.), the underlying technologies are largely irrelevant.

MAP 3.0 specifies operation only on EIA 802.4 LAN's, but many sites long ago committed to 802.3 or some other technology, and are unwilling for cost reasons to change. MMS, however, relies on the underlying layers to handle the details of maintaining full-duplex connections between two end points on a network. The intervening layers hide the details of the physical media and media access mechanisms, as shown in Figure 5.

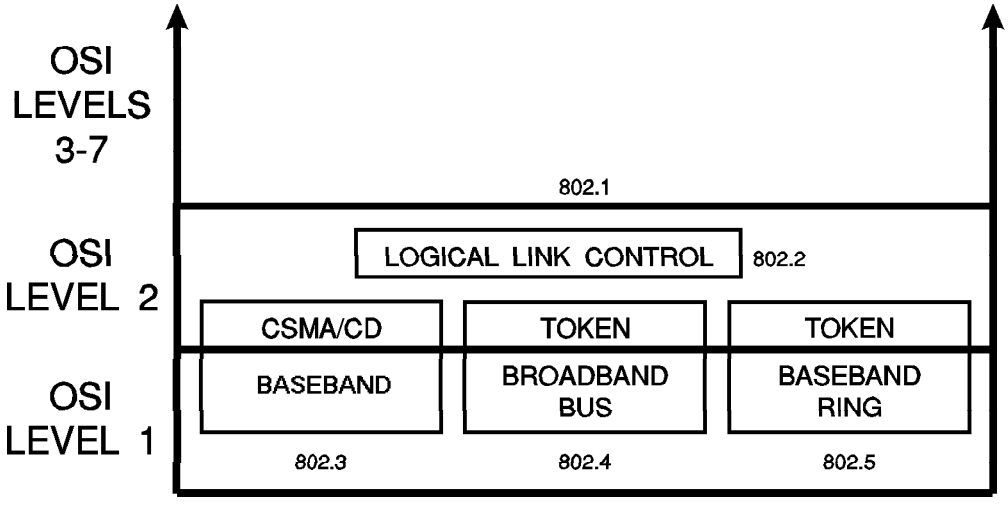

Figure 5: Low-level OSI Access Schemes

It is also worth pointing out that MMS is the only internationally accepted application-layer protocol specific to the manufacturing environment, and that MMS has been defined in a general enough way that it can be and indeed already has been implemented using other communications protocols, including TCP/IP, DECnet, and others, as shown in Figure 6. Those implementations are ad hoc, and as such are not guided by international standards in the way that MMS on an OSI stack is. The point is to show the essential independence of MMS from MAP.

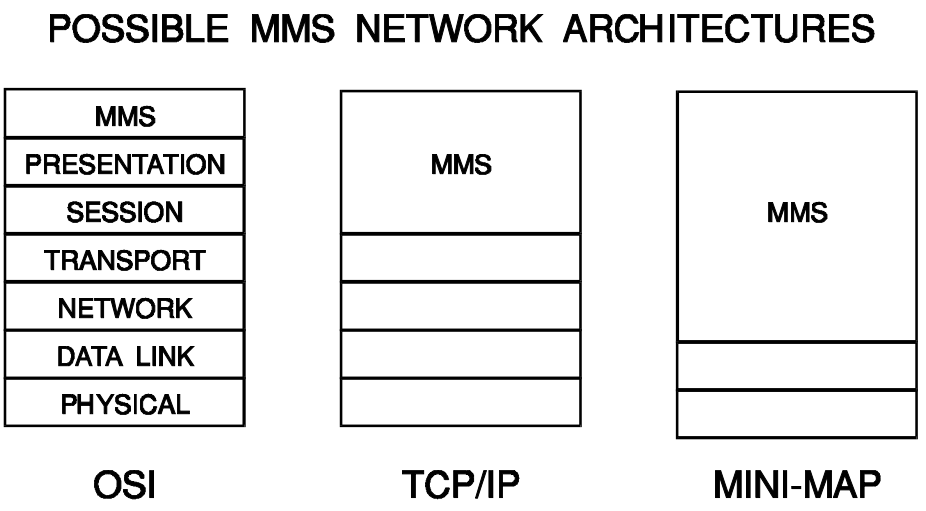

Figure 6: MMS Network Architectures

(TCP/IP and MINI-MAP architectures require ad-hoc agreements to be operational)

Where does MMS fit in the OSI scheme of things? In layer seven, the application layer. More properly, MMS itself consists of transitory messages that travel on the network. What does appear in layer seven is an Application Program Interface (API). (An MMS Interface or MMSI is often referred to as an API for network architectures utilizing MMS). The API consists of a library of routines that provides services to an application program through the API, as shown in Figure 7.

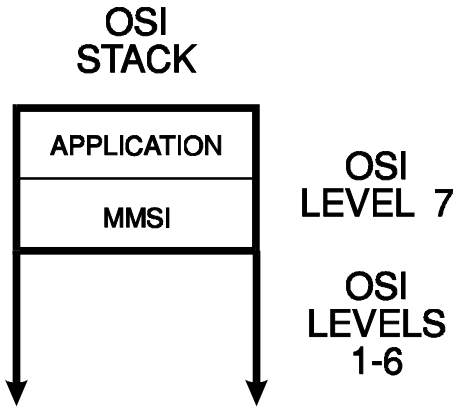

Figure 7: MMS Interface

Although MMS is defined by an international standard, there is no corresponding standard for the MMSI. This means that, in a given situation, MMS messages traveling on the network would be identical across different vendors' systems, but the method of sending and receiving those messages might differ. For example, DEC's API differs from IBM's, which in turn differs from Sisco's. This means that application programs are not directly portable across competing systems.

For users, this is an unfortunate situation; but it gives vendors a way of differentiating their products. There has been some talk of either standardizing an MMS API or of developing a

generic interface to the vendor-specific APIs, but competitive implications may prevent the development and acceptance of a common MMS interface. This should not be a problem for users with no interest in cross-vendor application portability.

Now that we know where MMS fits in the OSI world, let's see what MMS consists of internally.

Inside MMS

Because of the long gestation period of MMS, its architects have been able to design it using today's leading software technologies, including the client-server connectivity model, as well as an object-oriented structure.

Client-server Model

The client-server model is defined by an association between two nodes, a client and a server. Generally speaking, the server has something that the client wants. To get it, the client sends a message to the server, which in turn sends a message that contains the desired information.

For example, a control computer (the client) in a factory needs to obtain status information from an NC machine (the server). An application program running on the control computer sends a Status Request message. The OSI low-level (layer six and below) functions transfer the message to the node specified as the server--the NC machine. Its low-level OSI functions ensure that the message was transmitted correctly and that it is in the correct format, and pass it on up the OSI stack to the NC machine's control program by way of MMS. The NC machine's control program obtains the desired status information, then hands it off to MMS to process. MMS sends it back down the local OSI stack, across the network, back up the stack at the control computer, and to the application program running there. The application program is then free to raise an alarm, force a system shutdown, continue processing, or perform some other action, as required.

It is important to understand that a Status Request message is a Status Request message is a Status Request message, and any MMS-compliant NC device or robot or sensor or PLC can respond to it in a meaningful manner. That consistency provides a great productivity gain for applications programmers, who are thereby relieved of having to learn the idiosyncrasies of every device on the network. That consistency also helps reduce maintenance and upgrade costs, which can range as high as 50% or even more of the cost of an installation.

MMS Objects

Each server application is built around an abstract object called a Virtual Manufacturing Device (VMD). A VMD models the external behavior of an actual device--an NC machine, a robot, etc. In addition, a VMD provides the means by which a client application can get at the functionality contained within the actual device.

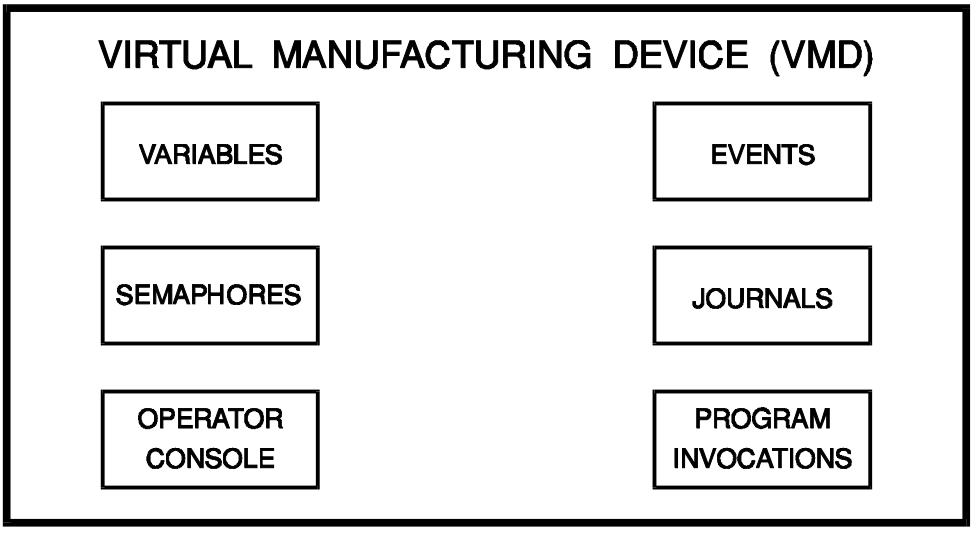

Figure 8: Virtual Manufacturing Device Model

As shown in Figure 8, a VMD also contains a set of objects that help coordinate access to the functionality represented by the object: simple and complex variables, semaphores, activity journals, program invocations, and operator input/output. Table 2 provides a brief description of each type of object.

Table 2—VMD Objects

Object	Description
Domain	A subset of the capabilities of a VMD.
Journal	A repository for archiving system information.
Operator Station	The device console of an actual machine.
Program invocation	An executable program and a domain.
Semaphore	A tool for synchronizing competition among several devices for a specific resource.
Variable	A path by which access can be obtained to a value or set of values. May consist of simple types, structures, or arrays.

MMS Services

A VMD consists of a set of objects that represents a real manufacturing device. MMS provides a rich set of 84 services for manipulating those objects. These services allow a client application to start and stop programs on the server, obtain information about the status of the server, access variables and semaphores, and communicate with the operator. Table 3 summarizes the MMS services by category.

Table 3—MMS Service Categories

Service	Description
Context Management	Establish and administer connections between nodes.
Domain Management	Create domains by reserving VMD capabilities, naming them, uploading and downloading binary images.
Event Management	Allow a client to define and manage server events by defining event conditions and event actions.
File Management	Simple file transfer/management.
Journal Management	Record and retrieve chronological information including events, variables, operator comments.
Operator Communication	Communicate with remote operator stations; includes input and output services.
Program Invocation Management	Bind, name, and control execution of a collection of domains by way of start, stop, resume, rest, kill, and delete commands.
Semaphore Management	Create and access semaphores to synchronize, control, and coordinate shared resources.
Variable Access	Define, read, and write remote variables.
VMD Support	Provide information about a device, including manufacturer, model number, status, available objects and resources.

Extensions and Companion Standards

MMS defines a generic set of objects and services that apply to a broad range of intelligent manufacturing devices. However, the MMS architects recognized that there would undoubtedly be situations in which the generic services and objects could not provide adequate performance. For that reason, provision was made to allow for extensions to the basic services. These extensions allow a vendor to customize and add value to its product.

Several industries have already identified desired extensions to the MMS core. These extensions are defined in Companion Standards to the basic MMS standards, and are administered by appropriate trade and standards associations, as shown in Table 4.

Table 4--Companion Standards

Robots/controllers	Robotics Industries Association (RIA)
Numerical controllers	Electronics Industries Association (EIA)
Process controllers	Instrument Society of America (ISA)
Programmable logic controllers	International Electrical Commission (IEC)

Section Summary

MMS provides a powerful communications protocol for monitoring and controlling disparate manufacturing devices. MMS grew out of the MAP project at General Motors Corporation, but soon embarked on a separate evolutionary path. Whereas MAP came to be viewed as one instance of the OSI network model, MMS gradually came into its own as a fully independent, internationally supported application-layer standard that may be implemented on any OSI-compliant network, and other networks as well.

Now that we have an understanding of the historical and technical issues, let's go on to view the place of MMS in industry.

MMS in the Big Picture

Viewed in a larger context, MMS is one component of a complex communication system. By itself, MMS can no more enable communication between several devices than an isolated telephone hand piece can enable conversation between two people. For MMS to achieve widespread support, the following are necessary.

* A broad spectrum of MMS products from multiple vendors
* Interoperable MMS hardware from multiple vendors
* Portable application software
* Migration strategies for adopting MMS

Vendor Support

Vendors in all spheres have announced support for MMS.

For example, virtually all major computer vendors have announced products that directly support MMS, including Digital Equipment Corporation, Hewlett-Packard, and IBM.

Many device vendors have also announced MMS products: Texas Instruments, Allen-Bradley, Mitsubishi, GE-Fanuc, Square-D, Moore Products Co., Computrol, and many others.

And several software houses and integrators (Savoir, Sisco, Retix) are providing low-level drivers and other MMS assistance to hardware vendors.

Other types of vendors also support MMS. Lotus Development Corporation, for example, has demonstrated an add-in for 1-2-3 that allows a spreadsheet to communicate with manufacturing devices through an @FACTORY function via MMS. Also, US Data's Factory Link provides an MMS interface through MAP.

Interoperability

Ideally, users should be able to make purchase decisions based on functional requirements, not compatibility issues. For example, if a user has an installed base of PLCs from Vendor A running on a network from Vendor X, he should not have to worry about network compatibility if he wants to upgrade to PLCs made by Vendor B. Users may also wish to freely mix and match devices from several vendors on the same network.

Both cross-vendor upgrades and multi-vendor solutions are possible today, but users must take responsibility for ensuring compatibility. On the positive side, some computer/device vendor pairs do provide definite interoperability within the limits of their product lines: e.g., DEC/Texas Instruments, IBM/Texas Instruments, DEC/Allen-Bradley.

Interoperability will most likely be market-driven: If users need specific devices from specific vendors to work together, and if vendors perceive a significant market opportunity, it is expected that vendors will respond accordingly.

Software Portability

A user might wish to port a custom application from IBM to DEC hardware. Or a software house might wish to develop an application for several hardware platforms.

Unfortunately, whereas there has been significant progress in interoperability, the lack of a standard MMS API limits inter-platform application portability. No standards currently exist for MMS API's, nor has any standards committee accepted responsibility for addressing this issue. (MAP 3.0 contains a description of an MMS API in the appendix, but this is not a standard). Users and software houses want application portability, and several vendors have indicated a willingness to develop products conforming to API standards, so it is likely that development of these standards will begin in the near future. Vendors view their APIs as a means of product differentiation, so until API standards exist, custom interfaces will be needed for inter-platform application portability .

Within a single manufacturer's product line, portability should be better. For example, if a user upgrades to a more powerful model, minimal application-level software changes should be required.

In addition, Savoir's FLEXIS API promises to provide some relief by providing a consistent MMSI on a variety of run-time platforms.

Migration

MMS was designed to facilitate control of disparate programmable manufacturing devices. As such it requires little specific functionality from those devices, but rather a way of controlling and responding to situations common to nearly all programmable devices: starting and stopping programs, initializing variables, reporting status.

For that reason, it is feasible to retrofit the large existing base of programmable devices with an extra communications layer that responds to MMS messages. Of course, devices built from the ground up with MMS compatibility in mind will ultimately provide enhanced capabilities. However, retrofitting existing installations eases short-term costs, eases the technical transition, and helps justify adoption of MMS over the long haul.

Another migration issue is software expertise. MMS provides an extremely rich set of services. Even though most users will use only a small subset of those services, becoming proficient in MMS programming can take several months. The long-term advantage, of course, is that learning MMS is a one-time affair; by using MMS-conformant products, it will be unnecessary to learn a new communications protocol every time a new device is released.

MMS Versions

For sites contemplating a new installation of MMS, this is not an issue.

However, some sites installed an interim version of MAP/MMS that was primarily designed for demonstration purposes at the Enterprise Networking Event (ENE) in 1988. It appears that there will be no new installations of this version. The NIST OIW MMS SIG (National Institute of Standards and Technology OSI Implementor's Workshop MMS Special Interest Group) has defined an implementor's agreement that addresses compatibility between the ENE '88 and the ISO 9506 versions.

MAP and MMS

As should be clear from this paper, MMS was born of the MAP Task Force, but separated from it early on, and is now a completely independent standard administered by an accepted international organization, the ISO.

Conformance Testing

There has been talk of establishing an independent body to perform conformance testing, but vendors indicate that problems must be worked out on a case-by-case basis. It is of course in a vendor's best interest to resolve any problems that may arise.

Benefits and Risks of MMS

Right now MMS offers significant benefits for connectivity and interoperability among programmable manufacturing devices. However, MMS is still at an early stage in its market acceptance. Consequently, there are risks involved with its adoption. This section outlines the significant benefits and risks.

Benefits

The biggest benefit to be gained from adopting MMS is better communication among manufacturing devices. This communication will allow the overall process to be monitored and controlled more closely, thereby increasing bottom-line productivity.

The overall benefit will be served by a number of subsidiary benefits, as follows.

* Software application development will require less detailed knowledge of disparate devices, because the devices will respond to MMS messages in standard ways. Ultimately, this standardization will reduce development costs.

* As vendors come on-line, use of custom (expensive, hard-to-maintain) software and hardware will decrease.

* Eventually, software maintenance costs will decrease due to plug-and-play capability. For example, upgrades will not require rewriting custom interface software.

* Vendor dependencies will be minimized.

* Use of a standardized network model allows better overall corporate integration.

* Smaller vendors can provide innovative products because there will be single standard.

* The MMS specification includes core functions and allows for future growth.

* Application programs will become more portable. When a device is upgraded or replaced by a more-powerful model, software migration will be minimized.

Risks

Currently MMS is suffering from a lack of critical mass. Although MMS has achieved international standardization, it is still relatively new, and users are unaware of the benefits it can provide, so there are a limited (albeit growing) number of MMS products and services.

* Lack of a standard MMS API is a significant drawback to MMS. Without a standard API, cross-vendor software development involves much duplication of effort.

* Purchase decisions are difficult due to lack of agreement on what conformance to the MMS standard entails. Users must decide based not only on functional requirements, but technical compatibility as well.

* Short-term costs of adopting MMS are likely to be higher than using some other technology. However, as adoption of MMS increases, hardware volume will increase and prices will fall.

* Short-term costs of adapting present systems for MMS compatibility may be high.

* Short-term costs of building new systems may be high due to lack of critical mass.

* Companion standards are still developing, so specialized needs might not be met right away.

* Performance of proprietary systems will most likely outperform MMS in the short term.

* Software development costs may be high until users become proficient at building MMS applications.

Conclusions

As automation spread throughout manufacturing during the 1960s, 1970s, and 1980s, it became increasingly apparent that automating isolated machines was only a first step toward increased competitiveness and profitability. The next step was for automated manufacturing machines to learn to communicate, both with one another and with the rest of the enterprise. The question now is not whether but how to achieve fully computer-integrated manufacturing (CIM).

MMS grew to meet the needs of CIM. MMS originated from work done by the MAP Task Force at General Motors, but has evolved independently since 1985. MAP is a set of specifications for a complete seven-layer OSI environment appropriate for operating a manufacturing plant. MMS, by contrast, attacks just one part of the problem: a standard protocol for inter-device communication.

MMS is a viable, internationally accepted, vendor-independent communications standard, and it is quickly attaining critical mass, achieving the support of virtually all major computer and control vendors: Digital Equipment Corporation, Hewlett-Packard, IBM, Allen-Bradley, Texas Instruments, and many others. Each of these companies has announced and delivered MMS products, and more are on the way.

MMS is suitable for both continuous and discrete manufacturing, and it is capable of being extended to meet special needs (robots, NC machines, sensors, controllers).

MMS offers significant long-term benefits, most of which derive from reduced software costs due to the use of a standard message protocol across disparate manufacturing devices.

The two biggest uncertainties about MMS are hardware interoperability and application portability. Ideally, users would be able to plug and play hardware from different vendors, and port applications across platforms with little or no redesign. Realistically, hardware interoperability has been achieved, but full cross-vendor application portability remains to be achieved.

In sum, MMS provides a bridge to both the past and the future, thereby allowing users to upgrade existing equipment, as well as to take advantage of powerful new technologies as they are developed.

References

EIA Standard 511, ANSI/EIA-511-1989

U.S. Government Open Systems Interconnection Profile (GOSIP), Version 1.0, June 1988

ISO/IEC 9506

"MAP Programmable Devices Command Language Simplifies Control Communications," Melvin L Hagar Jr., Texas Instruments, Industrial Systems Division.

"Impacts of MMS on Future Controllers," Melvin L Hagar Jr., Texas Instruments, Industrial Systems Division.

"Manufacturing Message Specification: Achieving Device Independence on the Factory Floor," M. A. Huras, S Bourbonnais, IBM Canada.

"Manufacturing Message Specification Interface (MMSI): The New Industry Standard For Developing Portable CIM Applications," M. A. Huras, S Bourbonnais, IBM Canada.

"Demystifying MMS and its Application Programming Interface," S. Vella, S. Bourbonnais, IBM Canada.

"MMS Application Development Using Object-Oriented Principles," R. Chan, P. Hodaie, S. Vella, IBM Canada.

Carrier-Band Network Handbook, "Manufacturing Message Service," (Chapter 6), Relcom, Inc. 1987.

"HP MAP 3.0 Manufacturing Message Specification/800," Hewlett-Packard Journal, August 1990, Peter A. Lagoni, Christopher Crall, and Thomas G. Bartz.

Manufacturing Automation Protocol, State-of-the-Art Study, December 1988, National Center for Manufacturing Sciences, Ann Arbor, MI.

"MAP Gradually Building Marketplace Acceptance," Paul Korzeniowski, Software Magazine, October 1990.

"MMS is the Key Emerging Interface Standard for Distributed Plant Applications," The Yankee Conveyor, March 1989.

"An Overview of the Manufacturing Message Specification," Ralph Mankiewicz, Presented at the North American MAP/TOP User's Group Meeting, September 1989.