Full MAP에서의 MMS 구현

°
최 엽, 하 정현, 채 영도

한국전자통신연구소, 자동화시스템 연구실

An Implementation of MMS on Full MAP Environment

Yobb Choi, Jeong Hyeon Ha, Young Do Chae
Automation Systems Section
Electronics & Telecommunications Research Institute

Abstract

Manufacturing Message Specification(MMS) was settled as an International Standard by International Standardization Organization(ISO). MMS is being accepted throughout the world as a solution to communications among multi vendor factory floor environments. This paper describes an implememntation of MMS which operates on Application Layer of Open Systems Interconnection(OSI) 7 layer model. MMS was implemented on MS-DOS in a personal computer environment.

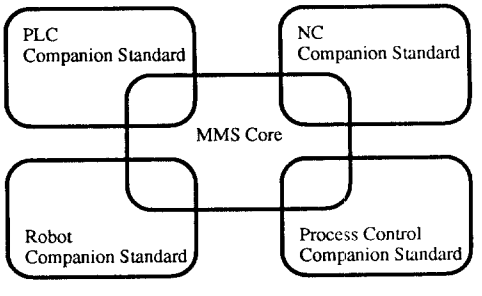
I. 서론

1970 년대까지는, 공장 환경의 자동화 시스템들간의 통신은 각 자동화 기기 메이커들이 나름대로 제공하는 통신 방법이나 프로토콜 들이 사용되고 있었다. 이처럼 난립하고 있는 여러 자동화 장비 생산 업체들의 통신 방법은,기업의 생산 활동에 있어 궁극의 목표로 되어있는 CIM (Computer Integrated Manufacturing)을 실현하는데 있어서 큰 장애 요인으로 판명되었다. 1980년대 초 General Motors를 중심으로 한 Manufacturing Automation Protocol(MAP) 개발 활동은 이러한 장애를 극복하기 위한 노력으로, 전 세계의 많은 기업체들의 지지를 받아 국제적인 de facto 표준으로 자리를 잡고있다[1].

MMS는 초기의 MAP 규격에서 제시되고 있었던 Manufacturing Message Format Standard(MMFS)를 전신으로 하고 있다. MMFS는 GM사 내에서 사용되고 있었던 장비간의 통신 프로토콜이었으나, 이를 국제적인 규격으로 제정하기에는 부적합하였기 때문에, 이를 바탕으로 하여, MAP 개발 활동의 일환으로 MMS를 개발하게 되었다. MMS는 수년간 국제적으로 산업 자동화 분야의 많은 전문가들의 집중적인 협력하에 개발져 어, MMS와 관련된 여타 작업들까지 완결되어 가고 있는 단계에 있다. ISO에서는 1990년에 MMS를 International Standard(IS) 9506으로 제정하여 국제 규격으로 만들었다[2],[3].

MMS는 (그림 1)에서와 같이, 서비스 스펙 (Service Specification) 과 프로토콜 스펙 (Protocol Specification)의 두 부분으로 구성되는 이론 바, 코어 스펙(Core Spec)과 프로그래머블 로직 콘트롤러(PLC), 수치 제어 기기(NC machine), 로보트, 프로세스 콘트롤 시스템 등과 같은 각각의 자동화용 장비들의 통신에 관련된 부수적인 스펙인 컴패니언 스탠다드 (Companion Standard)로 구성되어 있다[4],[5],[6],[7].

MMS는 OSI의 7 계층 참조 모델에 근거하여 작성되었으며, 7 계층의 응용 계층에 위치하여 구현되고 사용되는 것을 의도하고 있다. MMS는 응용 계층의 Association Control Service Element (ACSE) 의 상부에 위치하여 유저 프로그램과의 인터페이스를 갖는다.

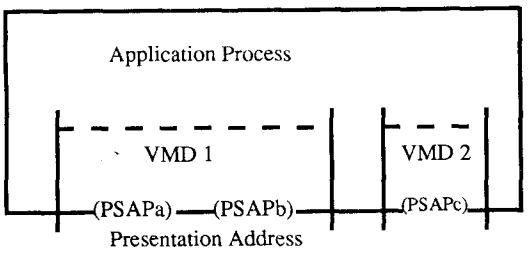


(그림 1) MMS Standard

MMS는 공장 환경에 있어서의 통신 서비스를 애매 모호함이 없이 표현할 수 있도록, 공장의 생산 장비의 일반화된 모델을 Virtual Maufacturing Device (VMD)로 정의하여, VMD를 구성하고 있으며 각종 통신 서비스의 대상이 되는 개념을 Object라고 하여 정확히 정의하고 있다.

따라서 ISO 9506 MMS는 VMD와 여러 Object들을 정의하고 이들의 사용 방법과 처리 방법을 기술하고 있는 셈이 된다. 이에 따르면 VMD는 (그림2)와 같이 OSI 환경에 들어 있다.

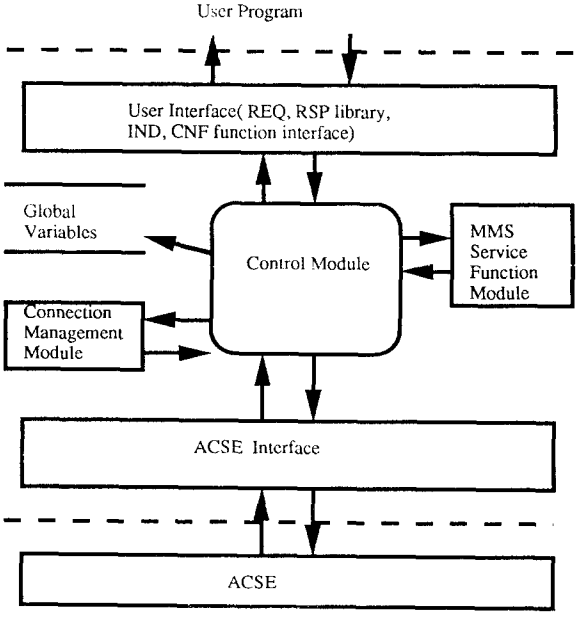
ASSMMS는 ISO 9506 MMS 에 따라 MS-DOS 상에서 MicroSoft-C를 사용하여 구현된 시스템이다.



(그림 2) VMD 와 OSI 개념

II. 구현 구조

ASSMMS의구현 구조는 (그림 3)와 같이 User Interface, Global variables, Connection Management Module, Contol Module, MMS Service Function Module, ACSE 인터페이스들로 구성되어 서로 interaction하는 형태로 되어 있다.



(그림 3) ASSMMS의 구현 구조

1. ACSE 인터페이스

MMPM은 OSI 응용 계층의 Association Control Service Element와 표현 계층(Presentation Layer) 서비스를 사용하게 되어 있다. 따라서 MMS의 Protocol Data Unit(PDU)는 각각 (그림 4) 와 같이 하위 계층에 매핑된다. 다만, MAP 2.1 스펙은 당시의 CASE(Common Application Service Element)의 커널을 지정하고 있었고, 표현 계층이 없는 채로 바로 세션 계층에 매핑하게 되어 있었다. 또한 MMS의 Abort 서비스는 PDU를 정의하지 않고 있으며 바로 ACSE의 A-Abort로 매핑된다.

MMS PDU	ACSE 또는 Presentation 서비스 프리미티브
Confirmed-RequestPDU	P-Data request, indication
Confirmed-ResponsePDU	P-Data request, indication
Confirmed-ErrorPDU	P-Data request, indication
Unconfirmed-RequestPDU	P-Data request, indication
RejectPDU	P-Data request, indication
Cancel-RequestPDU	P-Data request, indication
Cancel-ResponsePDU	P-Data request, indication
Cancel-ErrorPDU	P-Data request, indication
Initiate-RequestPDU	A-Associate request, indication
Initiate-ResponsePDU	A-Associate response, confirm
Initiate-ErrorPDU	A-Associate response, confirm
Conclude-RequestPDU	P-Data request, indication
Conclude-ResponsePDU	P-Data request, indication
Conclude-ErrorPDU	P-Data request, indication

(그림4) MMS PDU의 매핑

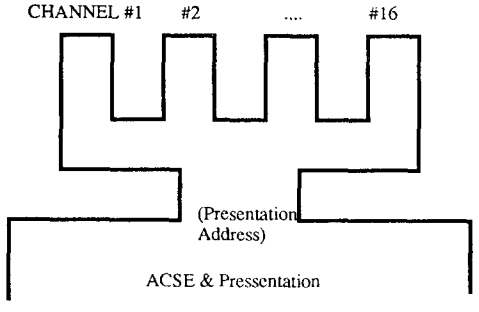
2. Connection Management

MMS의 Environment & General Management 서비스에 해당하는 기능을 하는 모듈로서 본격적인 MMS 통신의 이전에 확보되어 있어야 할 통신 상대방의 기본적인 정보를 주고 받게 해준다.

ASSMMS는 (그림 5)에서와 같이 한 프레젠테이션 어드레스 상에서, 논리적으로 독립된 통신 루트를 만들어 다중의 통신을 가능케 하는 채널 개념을 구현하고 있다. 통신 채널은 16개까지 가능하게 하고 있고, 시스템의 자원 형편에 따라 선택할 수 있도록 하였다.

Connection Management 모듈은 다중의 통신 채널에 있어서 각각 통신 상대방의 Application Reference 네임 또는 통신 능력 등에 관한 정보를 유지하고 있다. 이 모듈은 통신 상대방에 대한 정보를 저장하고 있는 다음과 같은 데이타 구조를 갖고 있으며, MMS VMD에 관련된 데이타들도 갖고 있다. 추후 컴패니언 스탠다드의 구현이 추가되면 이에 따른 통신 상대방과의 negotiation을 수행한다.

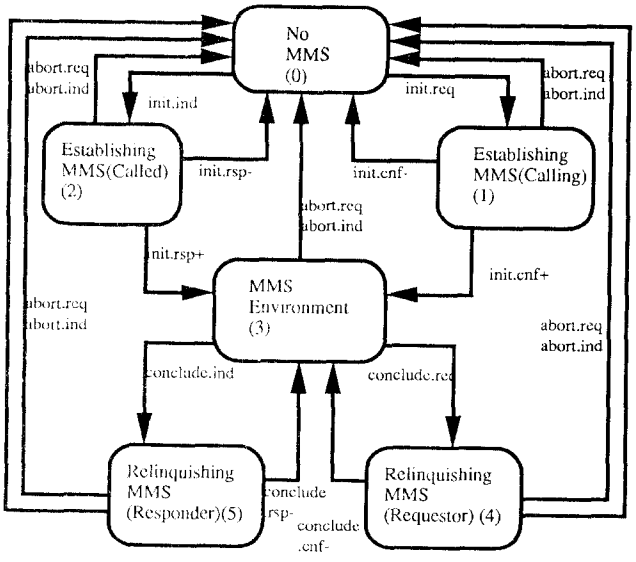
```
STRUCT {
        CHANNEL STATE;
        MMS CONTEXT;
        LOCAL APPLICATION REFERENCE NAME;
        REMOTE APPLICATION REFERENCE NAME;
        VMD STRUCTURE;
        DOMAIN STRUCTURE;
        MMS VERSION;
        INITIATE INFORMATION;
            COMMUNICATION CAPABILITY;
            SERVICE CONFORMANCE BULDING BLOCK;
            PARAMETER CONFORMANCE BUILDING
BLOCK;          };
```



(그림 5) ASSMMS 의 채널 개념

2.1 State diagram

ASSMMS의 state는 ISO 9506에서 규정하고 있는 (그림 6)과 같은 상태의 state transition을 보인다. MMS의 모든 서비스는 이 state의 MMS Environment 상태에서 실현된다.



(그림 6) MMPM의 state diagram

2.2 State table

(그림 7)은 MMPM의 state table로서 안은 MMPM의 action과 그 후에 들어갈 상태를 보이며, 빈 칸은 에러인 이벤트이다.

state / event	No MMS (0)	Establishing MMS(Calling) (1)	Establishing MMS(Called) (2)	MMS Environment (3)	Relinquishing MMS (Requestor) (4)	Relinquishing MMS (Responder) (5)
initiate_req	TxInitReq Frame (1)					
conclude_req				TxConcReq Frame (0)		
abort_req			TxAbrtReq Frame (0)	TxAbrtReq Frame (0)	TxAbrtReq Frame (0)	TxAbrtReq Frame (0)
initiate_rsp+			TxInitRsp+ Frame (3)			
initiate_rsp-			TxInitRsp- Frame (0)			
conclude_rsp+						TxConcRsp+ Frame (0)
conclude_rsp-						TxConcRsp- Frame (3)
initiate_ind	RxInitInd Frame (2)					
conclude_ind				RxConcInd Frame (5)		
abort_ind		RxAbrtInd frame (0)	RxAbrtInd frame (0)	RxAbrtInd frame (0)	RxAbrtInd frame (0)	RxAbrtInd frame (0)
initiate_cnf+			RxInitCnf+ Frame (3)			
initiate_cnf-			RxInitCnf+ Frame (0)			
conclude_cnf+					RxConcCnf+ Frame	
conclude_cnf-					RxConcCnf- Frame	

* TxInitReqFrame: Transmit Initiate Request Frame
TxConcReqFrame: Transmit Conclude Request Frame
TxAbrtReqFrame: Transmit Abort request Frame
TxInitRsp(+)Frame: Transmit Positive Inititae Response Frame
TxInitRsp(-)Frame: Transmit Negative Initiate Response Frame
TxConcRsp(+)Frame: Transmit Positive Conclude Response Frame
TxConcRsp(-)Frame: Transmit Negative Conclude Response Frame
RxInitIndFrame: Receive Initiate Indication Frame
RxConcIndFrame: Receive Conclude Indication Frame
RxAbrtIndFrame: Receive Abort Indication Frame
RxInitCnf(+)Frame: Receive Positive Initiate Confirmation Frame
RxInitCnf(-)Frame: Receive Negative Initiate Confirmation Frame
RxConcCnf(+)Frame: Receive Positive Conclude Confirmation Frame
RxConcCnf(-)Frame: Receive Negative Concldue Confirmation Frame

(그림 7) MMPM의 state table

3. Global Variable

MMS 의 각종 서비스에 고유한 여러 데이타 구조 변수들로서, MMS 응용 프로그램 인터페이스 스펙 (Application Program Interface Specification) 에서 제시하는 C-언어 Binding에 따른 데이타 구조로 되어 있다. 또한 여기에는 네트웍 인터페이스 보드 등 기타 네트워킹에 관련된 일반 변수들이 포함되어 있다.

4. Control Module

ASSMMS의 전반적인 동작을 관리하는 모듈로서 MMPM(Manufacturing Message Protocol Machine)의 역할을 수행한다. System function과 VMD related function, 에러 처리 부분으로 구분한다.

4.1 System function

가. 인터페이스 function

ASSMMS의 전반적인 초기화, 시스템 자원의 해제 등을 수행한다.

- MAP 인터페이스의 초기화을 수행한다.
- MAP 프로토콜 인터페이스의 초기화을 수행한다.
- 채널 인포메이션 array의 초기화.
- PEND 큐의 초기화.
- IND 큐의 초기화.
- 통신 종료시 allocate되었던 memory의 release.

나. 데이타 송수신 function

네트웍으로 나가는 메시지와 네트웍으로부터 올라오는 메시지를 처리한다. 유저는 주기적으로 이 function을 call하여 메시지를 보내거나 받는다.

다음과 같은 3가지의 sub function으로 구성된다.

- 하위 계층과의 메시지 송수신 모듈

```
if ( IND PDU )
    parse PDU;
    fill in IND structure;
    put it in queue for indication processing;
if ( CNF PDU )
    parse PDU;
    search PEND queue, and pair it;
    put it in queue for confirmation processing;
```

- Indication 서비스 function

```
read from IND;
call appropriate user indication function;
```

- Confirmation 서비스 function
  read from PEND;
  call appropriate confirmation function;

4.2 VMD related function

ASSMMS가 VMD로 사용될 수 있도록 제반 환경을 만들어 주는 부분으로 다음과 같은 기능을 하는 function들로구성된다.

- MMS 유저가 MMS variable access나 semaphore service의 server로 동작시킬 수 있도록 한다. MMS 유저가 MMS object들을 name으로 access할 수 있도록 한다.

- 해당 호스트 시스템에서 사용될 수 있는 VMD identifier를 만든다. 이 VMD에 등록될 AE가 있음을 알린다.

- VMD identifier를 invalidate시킨다. VMD identifier에 연결된 모든 AE, semaphore, variable, variable list, named type과 variable 은 모두 deactivate하거나, delete한다.

- VMD와 관련된 모든 object를 완전히 제거하고, VMD를 deactivate된 상태로 둔다.

4.3 에러 처리

ASSMMS의 exception 처리는, Function execution 에러와 communication 에러의 두가지 class로 나뉜다.

가. Function execution 에러

유저 application에게 available한 ASSMMS function은 그 function이 잘 수행되었는지를 리턴해 준다. 포인터를 리턴해 주는 function의 경우에 zero(0)나 NULL의 포인터는 에러를 가리킨다. 포인터를 리턴하지 않는 경우에는 그 리턴 값이 에러 코드이다. 리턴 값이 0이면 function이 제대로 수행된 것이고, 그렇지 않은 경우에는 그 값이 바로 에러 코드이다.

ASSMMS 의 리턴 값이 success를 가리키더라도, MAP interface board가 그것을 lower layer에 보내기 전에 reject를 할 수 있다. request function에 대해 이런 일이 일어 나면, response도 받을 수 없을 뿐만 아니라, transport layer가 PDU를 받지 않았기 때문에 timeout 에러도 받을 수 없다. 이러한 에러의 중대성을 감안하여 ASSMMS는 자동적으로 exception indication function을 부른다. 이 때 유저는 association을 끊을 수 있다.

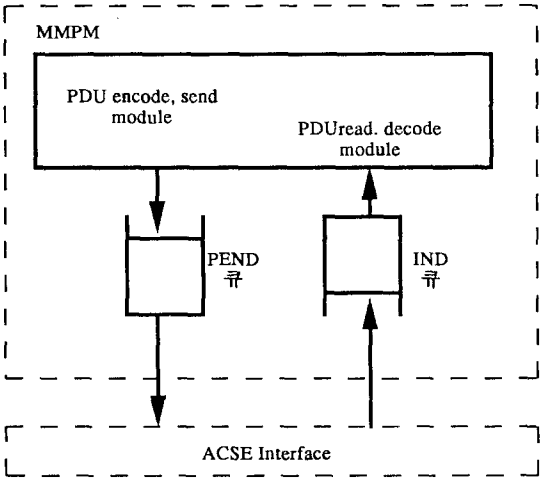
나. Commuication 에러

ASSMMS의 communication 에러는 보통 PDU parse 에러이다. indication에서 parse 에러가 생기면, reject response가 보내진다. confirmation에서 에러가 생기면, ASSMMS는 그 confirmation을 outstanding한 request와 match시키고 response 에러 코드를 set한다. ASSMMS는 에러와 reject PDU를 받아들이고, 가능하면 그것을 outstanding한 request와 pairing한다. message type이 판명되기 전에 parse 에러가 생기면 ASSMMS는 그 PDU를 무시한다.

4.4 큐 구조(Queue data structure)

유저가 xxxx_req function을 사용하여 서비스 request를 발행할 때, outstanding한 request를 추적할 수 있도록 pending queue data structure를 제공한다. xxxx_req function은 이 data structure에 대한 포인터를 return하게 되며, 유저는 필요에 따라 이 data structure의 내용을 읽거나 변경할 수 있다. 또한 이 control structure는 ASSMMS가 그 request에 대한 confirm을 받고, xxxx_CNF function이 invoke되고 return될 때까지 유효한 상태로 있게 된다.

indication이 받아들여질 때에는, ASSMMS가 PDU를 parse하여 그것에 대한 indication data queue를 assign한다. indication은, indication queue data structure에 대한 포인터와 함께 적당한 xxxx_IND function의 call을 통해 유저에게 전달된다. 이러한 indication function은 comm_serve function이 call될 때마다 수행된다. 이 iindication은, 유저 프로그램이 indication queue data structure를 가리키는 포인터를 가진 xxxx_RSP나 xxxx_err function을 call할 때까지 active한 상태로 남아 있게된다.



(그림 8) ASSMMS의 큐 구조

가. Pending Request Queue data structure

request가 발행된 후 그에 대한 confirmation을 처리할 때까지 쓰이는 data structure이다. 이 type은 start_MMS function을 call할 때에 그 element에 적당한 data가 store된다. 유저는 outstanding한 request를 추적하기 위해 이 control structure를 사용할 수 있다. xxxx_req가 call되면 이 structure를 가리키는 포인터가 return된다.

confirmation이 도착되면, ASSMMS는 이 structure를 탐색하여, 해당되는 request를 확인하고, u_xxxx_conf function이 수행되면 그 structure를 release한다. pending queue data structure PEND는 다음과 같이 정의된다.

```
PEND QUEUE STRUCTURE {
   QUEUE
     PREVIOUS QUEUE *;
     NEXT QUEUE *;
   CHANNEL NUMBER;
   CONTEXT;
   OPERATION CODE;
   INVOKE ID;
   CANCEL STATE;
   RESPONSE STATE;
   REQUEST INFORMATION;
   RESPONSE INFORMATION;
   };
```

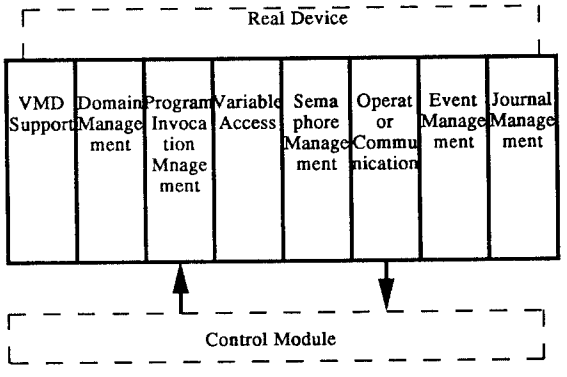
나. Indication queue data structure

indication을 처리하기 위한 data structure type으로서 MMS의 start시에 그 array가 할당된다.

```
PEND QUEUE STRUCTURE {
    QUEUE
      PREVIOUS QUEUE *;
      NEXT QUEUE *;
    CHANNEL NUMBER;
    CONTEXT;
    OPERATION CODE;
    INVOKE ID;
    CANCEL STATE;
    REQUEST INFORMATION;
    RESPONSE INFORMATION;
    };
```

5. MMS 서비스 Function Module

MMS 스펙의 각종 서비스의 프로토콜 머신을 구현하고 있으며, 프로토콜 데이타 유닛의 ASN.1 인코딩 및 디코딩을 담당하는 부분이다. (그림 9는 MMS 서비스들과 실제 디바이스들과의 관계를 보이고 있으며, 여기의 각 서비스 지원 모듈들과 실제 디바이스들의 기능과의 매핑이 추후 구현되어야 할 사항이다.

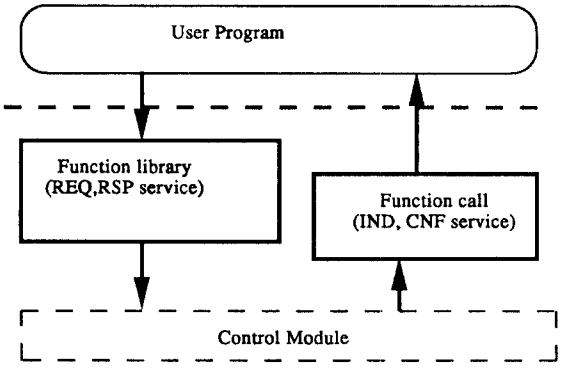


(그림 9) ASSMMS의 서비스 모듈

6. 유저 인터페이스

6.1 유저 인터페이스의 구조

ASSMMS의 유저와의 Interface는 (그림 10)과 같이 크게 Library function과 Interface function으로 구분된다. Library function은 시스템에 library 형태로 저장되어 있으며, 유저 program이 필요에 따라 call해서 쓰는 function이다. Interface function은 ASSMMS가 call하는 function으로서 유저가 정의해 주어야 하는 program이다.



(그림 10) ASSMMS의 유저 인터페이스

6.2. ASSMMS library functions (User -> MMPM)

MAP 3.0의 MMS Application Interface spec은 Client의 MMS inteface를 규정하고 있는 바, 본 스펙은 MMS Application Interface 문서의 naming을 충실히 따르고, 그 문서에 규정되어 있지 않은 서비스들은 적당한 delimiter를 사용하여 구별하도록 한다. 즉, 이하에 열거하는 서비스 프리미티브들을 중심으로 하고, 추가되는 서비스 function name은 " _REQ, _RSP" 를 덧붙여 구별한다. function은 기본적으로, 서비스가 전달되어야 하는 데이타 채널 번호와, 각 서비스에 고유한 데이타 구조 변수에 데이타를 그의 입력으로 하고 ,그 function의 리턴 값으로는 PEND 큐에 대한 포인터를 돌려 받는다. function의 형태는 다음과 같다.

```
Return value    *ssdxxx[z]*(channel, pointer)
    Input
        Integer  channel;
        service_specific_structure_variable *pointer;
    Return value
        PEND_queue *
```

6.3. ASSMMS interface functions (MMPM -> User)

이 function들의 name은 function name에 indication의 경우에는 _IND, confirmation의 경우에는 _CNF이 부가되는 형태이다. function의 형태는 다음과 같다.

```
Return value    *ssdxxx[z]*(ptr)
    Input
        IND-queue  *pointer;
    Return value
        VOID
```

6.4 Function naming 규칙

유저 Interface 의 MAP 3.0의 Application Interface에서제시하는 바와 같이 처음 여섯 문자까지 내에서 unique하게 되어 있다. 즉, 다음과 같은 형태를 가지게 된다.

ssdxxx[z]*

ss: service identifier

d: [language dependent] delimiter

xxx: function name의 unique part(3 문자까지)

[z]*: unique 할 필요가 없는 부분

또한 에러 name은 다음과 같은 형태로 정해진다.

ssExxx[z]*

xxx: 3 문자로 된 unique한 에러 name

Primitive들의 naiming은 다음과 같은 형태이다.

ss_***_ppp

ppp -> request: REQ

indication: IND

response: RSP

confirmation:CNF

*** -> 실제 프리미티브를 나타내는 letter

ASSMMS 에서 다루는 모든 이름들은 다음과 같은 원칙하에 정해졌다.

- 모든 constant는 대문자
- 모든 variable name은 소문자
- 모든 function name은 소문자
- 모든 struct/union/typedef은 첫째 letter는 대문자로 나머지는 소문자로, 단 acronym이나 word의 첫째 letter는 대문자로 할 수 있다. 다만, ASSMMS의 동작 환경이 MAP 3.0의 MMS Application Interface에서 지향하는 구현 환경과 다르기 때문에, function의 parameter 형태는 MMS Application Interface와는 다른 형태로 정해졌다.

III. 결론

본 MMS Implementation은 ISO 9506스펙을 최대한 충실히 반영하고자 한 시스템으로서, MMS의 Server와 Client를 모두 구현하고 있으며, Client측의 구현에 있어서는, MAP 활동의 중요한 작업결과 중의 하나인 MMS Application Interface Specification에도 가능한 한 충실히 따랐다.

다만, 이 시스템은 단일 태스크만을 동작시킬 수있는 MS-DOS상에서 구현되었기 때문에 자체로 많은 한계점을 가진다. 멀티 태스킹 환경하에서의 MMS 구현은 유저 프로그램에 더 많은 유연성을 주게 될것이다.

이 Implementation은 Full MAP에서의 MMS 구현의 국내 첫 시도중의 하나로서, 당연히 이 Implementation은 국제 표준 시스템으로서의 적합성 시험(Conformance Test)의 필요성을 제기한다. 국내에서의 적합성 시험의 방법론이 정립되고 이 Implementation이 시험된 연후에 멀티 벤더간의 상호 동작성(Interoperability Test)이 시험되어야 한다.

또한 이 시스템은 OSI의 응용 계층의 위에 동작하는 프로토콜로서, 초기 MAP 활동에서 중요시 하였던 자동화 장비간 통신의 실시간(Real-time)성이 고려되지 않았기 때문에 이에 대한 추가적인 분석과 그에 따른 개선을 필요로 한다.

MMS는 공장용 통신의 공통 부분이고, 실제 현장의 생산을 담당하는 프로그래머블 디바이스의 통신에 관한 요구사항은 MMS의 컴패니언 스탠다드가 규정하고 있으므로, 이 Implementation에의 컴패니언 스탠다드의 수용을 위한 노력이 계속되어야 한다.

IV. 참고 문헌

[1] Manufacturing Automation Protocol, Version 3.0, General Motors, Aug.1988.

[2] ISO 9506-1, "Manufacturing Message Specification, Part1: Service Specificatoin", Feb. 1990

[3] ISO 9506-2, "Manufacturing Message Specification, Part2: Protocol Specificatoin:, Feb. 1990

[4] ISO TC 184 SC5 WG2 N200, "Manufacturing Message Specification - Part 3: Robot Specific Message System", Jan. 1990.

[5] ISO TC 184 SC5 WG2 N201, "Manufacturing Message Specification - Part 4: Numerical Control Semantics for the Manufacturing Semantics for MMS Service and Protocol Standard", Feb. 1990.

[6] ISO TC 184 SC5 WG2 N208, "IEC SC65A WG6 TF7 (Coordinator 7) - Programmable Controller Message Specification", Feb 1990.

[7] ISO TC 184 SC5 WG2 N205, "Draft - IEC/ISO 9506 Manufacturing Message Specification - Part 6: Process Control Semantics for the MMS Service definition and Protocol Specification", Mar. 1990.

[8] ISO TC 184 SC5 WG2 N196, "FAIS Cell-Net Implementation Specification", Jun.1989.

[9] ISO 8824, " Information Processing Systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One(ASN.1)", Dec. 1987

[10] ISO 8825, " Information Processing Systems - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Syntax Notation One(ASN.1)", Dec. 1987.

[11] ISO DIS 8650, "Information Processing Systems - Open Systems Interconnection - Service Definition for Association Control Service Element", 1988.

[12] ISO DIS 8651, "Information Processing Systems - Open Systems Interconnection - Protocol Definition for Association Control Service Element", 1988.