

# ICCP를 사용한 전력센터간의 통신 프로토콜 구현

장 경 수<sup>†</sup> · 장 병 옥<sup>††</sup> · 한 경 덕<sup>†††</sup> · 신 동 렬<sup>††††</sup>

## 요 약

현재의 전력시스템은 컴퓨터와 통신기술의 발전으로 분산시스템인 EMS(Energy Management System)/SCADA(Supervision Control and Data Acquisition) 형태로 운영되어 전력의 생산, 전송 그리고 분배가 효과적으로 이루어지고 있다. 그러나 각 시스템은 언어, 운영체제 그리고 통신프로토콜이 서로 다른 제품으로 구성되어 시스템간에 데이터를 교환하는데 많은 어려움이 따르고 있다. 이러한 문제를 해결하기 위해서 미국 전력연구소는 전력제어센터간의 통신을 담당하는 ICCP(Inter-Control Center Protocol)라는 새로운 형태의 통신규약을 발표하였다.

ICCP는 자동화용 표준 통신규약인 MMS(Manufacturing Message Specification)를 응용계층의 하부 규약으로 지정함으로써 서로 다른 기종의 제어센터간의 원활한 통신을 지원한다. 본 논문은 ICCP의 특징과 MMS와 ICCP가 어떻게 상호 연관되는가를 밝힌다. ICCP에서 이용하는 MMS 라이브러리(library)의 86개의 서비스 중 일부 서비스를 구현한 후 이것을 이용하여 TCP/IP 환경 하에서 ICCP의 기본이면서 핵심적인 기능을 구현한다. 그 다음, ICCP 프로토콜을 이용하여 EMS간의 통신을 모델링하고, 전력센터간의 실제 데이터 교환을 윈도우 환경 하에서 구현하여 ICCP 프로토콜의 동작과 기능을 보여준다.

## Implementation of Communication Protocol between Control Centers using ICCP

Kyung-Soo Jang<sup>†</sup> · Byung-Wook Chang<sup>††</sup> · Kyung-Duk Hahn<sup>†††</sup> · Dong-Ryeol Shin<sup>††††</sup>

### ABSTRACT

Current power systems are distributed geographically and operated in the form of Energy Management System(EMS)/Supervision Control and Data Acquisition(SCADA) with the aid of computers and communications. Recently a variety of utilities have had interests in using information technology to bring the efficiency and low operational costs. There is also a trend to integrate the production, transmission, distribution and management/control of power into one and unified distributed system. To this end, Electric Power Research Institute(EPRI) announced a new standard communication protocol called Inter-Control Center Protocol(ICCP). ICCP specifies the use of Manufacturing Message Specification(MMS) for services required by ICCP in application layer and supports the communications between heterogeneous control centers.

This paper presents the characteristics of MMS, ICCP and their relationship. Furthermore, we implement the basic functional blocks of ICCP using MMS services under TCP/IP environments. Finally, we model a simple power system and apply the ICCP protocol to this system in a window-based scheme, and finally show the operation and validation of this protocol.

† 준 회 원 : 성균관대학교 대학원 전기전자 및 컴퓨터공학과  
†† 준 회 원 : 삼성전자(주) 이동통신개발팀  
††† 준 회 원 : 인포섹코리아(주)

†††† 정 회 원 : 성균관대학교 전기전자 및 컴퓨터공학부 교수  
논문접수 : 2000년 3월 27일, 심사완료 : 2000년 12월 1일

## 1. 서 론

1980년대 초반 미국의 General Motors에서는 자사의 공장에 있는 여러 업체에서 만들어진 장치들을 효율적으로 제어하여 생산성을 높일 수 있도록 실시간 데이터 교환이 가능한 MAP(Manufacturing Automation Protocol)을 발표하였다. MAP은 분산배치 되어있는 로봇(robot), PLC(Programmable Logic Control), NC(Numerical Control) 등을 하나의 네트워크로 연결하여 상호간 메시지를 주고받게 함으로써 통합생산라인을 효율적으로 구축하는데 그 목적이 있다[1]. MAP은 이후에 국제 표준화기구인 ISO(International Standard Organization)로부터 산업용 통신망 구축의 표준으로 정해졌다.

MAP은 응용계층 프로토콜로 MMS(Manufacturing Message Specification)를 지정하여 이를 사용함으로써 서로 다른 통신규약을 사용하는 기기들간의 통신이 가능하게 되었고, 이기종 기기간의 통신을 원활하게 할 수 있었다[2]. MMS는 이후 MAP과는 별도로 장치들간에 실시간 통신이 필요한 곳, 즉 우주항공, 석유화학, 빌딩자동화와 더불어 전력분야로 그 쓰임새가 확장되고 있다.

현재의 전력시스템은 컴퓨터와 통신기술의 발달로 많은 양의 데이터를 효과적으로 처리 할 수 있는 EMS(Energy Management System)/SCADA/(Supervisory Control And Data Acquisition)의 형태로 구성되어 실시간 데이터 처리와 데이터베이스 프로그램을 사용하여 많은 양의 데이터를 효과적으로 처리하고 있다. 그러나 각 시스템들은 업체들에 따라서 다양한 형태의 구조, 다양한 프로그래밍 언어, 다양한 운영체제 하에서 운영되므로 모든 시스템간의 통신을 지원하는 데에는 많은 어려움과 비용의 문제점들이 발생한다. 이것을 해결하기 위해 시스템에 독립적인 통신규약의 필요성이 대두되었다. 이에 전력분야에서는 1990년대 초 EPRI(Electric Power Research Institute)의 주도 하에 UCA(Utility Communication Architecture) 프로젝트를 시작하여 물, 가스, 전기 등의 산업에 대한 국제적인 표준을 제시하였다. 이에 따른 응용프로토콜의 표준으로 ICCP(Inter-Control Center Protocol)라는 프로토콜이 제안되었다[3].

ICCP는 서로 다른 각 제어센터간에 실시간 데이터, 제어데이터, 측정데이터 등의 데이터를 미리 정해진

규약에 따라 일정한 포맷으로 전송하기 때문에 그 표준을 만족하기만 하면 어떠한 전력센터 시스템 사이에도 통신이 가능하다. 또한 생산 기기의 제조업체에 대한 의존도를 감소함으로써 확장성이 뛰어나다는 장점을 가진다. 이에 반해 특정 응용분야에서는 자사고유의 통신규약이 ICCP를 능가할 수도 있으며 또한 단기적인 비용이 상승할 가능성이 단점으로 지적되고 있다. 하지만 장기적인 관점에서 보았을 때 ICCP가 가지고 있는 장점이 이러한 단점들을 압도할 것이다.

현재 미국 및 다른 국가들의 전력사업은 여러 사업자에 의한 다분화된 구조로 이루어져 전력회사간에도 전력의 판매, 공급 등이 이루어지는 복잡한 구조를 가진다. 이러한 환경은 국내의 전력산업에도 구조적 변화를 가져오고 있으며, 특히 전력계통을 운영하기 위한 통신프로토콜의 표준화는 절실하다고 하겠다. ICCP에 관한 내용은 현재 국내에서는 많은 연구가 이루어지지 않았지만 점차로 연구소와 학교를 중심으로 프로토콜 구현 및 응용이 확산되고 있다.

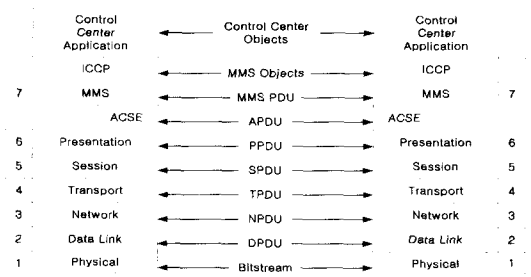
본 논문은 전력통신 프로토콜인 ICCP의 기본적인 면서 핵심적인 기능을 구현하고, 이를 간단한 모델에 적용하여 프로토콜의 동작과 유용성을 보여준다. 본 논문의 개략적인 구성을 살펴보면 다음과 같다. 2장에서는 ICCP와 MMS의 관계를 개략적으로 살펴볼 것이다. 3장은 MMS와 ICCP에서의 이벤트 관리(event management)의 차이점을 분석한다. 4장에서는 ICCP에서의 가상제어센터(Virtual Control Center : VCC)와 MMS의 가상제조기기(Virtual Manufacturing Device : VMD)를 비교한 후 ICCP의 동작과정을 설명한다. 5장은 ICCP 구현 부분으로 먼저 구현한 전체 시스템 구성 환경을 보인 후 구현된 ICCP 서비스의 일부를 보인다. 또한, EMS 클라이언트/서버와 SCADA로 시스템을 구성하고, ICCP를 이용한 전력센터간의 실제 데이터 교환을 위한 응용프로그램을 윈도우 환경에서 구현하고 그 실행화면의 일부를 보여준다. 결론은 마지막 6장이다.

## 2. ICCP 프로토콜

### 2.1 ICCP 프로토콜 구조

ICCP는 (그림 1)과 같이 OSI(Open System Interconnection) 7계층 중 최상위 계층에 존재하는 응용계층 프로토콜이다[3]. 응용계층 프로토콜이란 사용자 또는 제어센터 응용프로그램으로부터 메시지를 받아 하

부 계층으로 전달하고, 하부 계층으로부터 메시지를 받아서 사용자 응용프로그램으로 보내는 역할을 한다. ICCP는 응용계층 하부 프로토콜로서 MMS를 지정하고 있다. ICCP는 MMS의 86개의 서비스 함수 중 일부분을 호출하여서 사용하며 또한 MMS 객체를 사용한다. ICCP의 상위계층에는 각 제어센터의 응용프로그램들, 즉 EMS 프로그램 또는 SCADA 프로그램 등 각종 전력센터의 시스템들이 존재할 수 있다. MMS와 ICCP의 구현이 OSI 7계층의 참조모델 환경 하에서 지원되지만 본 논문에서는 TCP/IP를 기반으로 구현하며 이는 자동화 프로토콜이 실시간에 응용될 수 있도록 접차로 OSI 모델에서 TCP/IP 모델로 옮겨가고 있는 것과 같은 맥락이다.



(그림 1) ICCP 프로토콜 구조

2.2 MMS

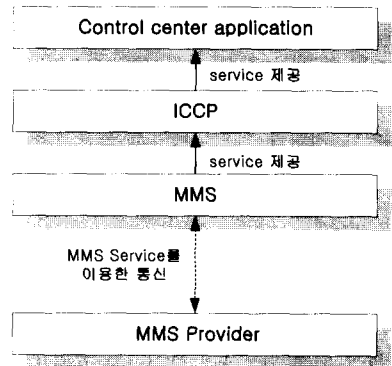
MMS는 서로 다른 제조사의 서로 다른 단위제어 기기간에 메시지 교환을 위한 ISO/OSI 참조모델 중 응용계층에 속하는 공장자동화용 프로토콜로 ISO/IEC 9506-1(이하 Part 1)[4]과 ISO/IEC 9506-2(이하 Part 2)[6]가 그 핵심 규약이다. 서비스 사양에 대한 문서인 Part 1은 가상제조기기, 네트워크 상에서 교환되는 메시지의 정의, VMD에 관련된 속성과 파라미터에 대해 기술한다[4]. Part 2는 프로토콜의 사양을 표현 계층(presentation layer)의 추상구문표현 1(Abstract Syntax Notation 1 : ASN.1)[10]으로 기술되어 있다[6]. MMS는 공장자동화용으로 개발되어 현재 본 논문의 전력 제어 외에도 여러 분야에 응용이 되고 있는 통신규약이다[1]. MMS를 사용함으로써 얻어지는 이점은 통합의 편리성, 설치비용과 유지보수 비용의 감소, 응용프로그램 이식 등의 장점이 있지만 무엇보다도 다기능 기기, 즉 서로 다른 프로토콜을 사용하는 기기들간의 통신을 가능하게 하는 상호운영성에 있다[4].

MMS는 가상제조기로서 각각의 디바이스를 표현하며, 가상제조기기는 속성과 동작으로 나누어져 디바이스의 모든 자원들을 나타낸다. 서로 다른 기기들을 가상제조기기로 표현함으로써 통신을 할 수 있는 환경을 제공한다. 이 통신규약은 클라이언트-서버 모델(client-server model)을 기본으로 하고 있으며, 클라이언트 시스템과 서버 시스템은 구조화된 데이터 블록인 PDU(Protocol Data Unit)를 이용하여 통신을 한다.

MMS는 Context, VMD, Variable, Domain, Program Invocation, Semaphore, Event, Journal 등 총 8개의 객체로 이루어져 있고, 객체들에 대해서 정의된 서비스는 86개에 이른다. 이들 객체와 서비스들의 일부분이 ICCP의 객체와 서비스들로 매핑되어서 사용된다. ICCP에서는 서비스를 동작과 행동으로 표현하고 있지만, 본 논문에서는 ICCP에서 정의한 동작과 행동에 대한 용어를 서비스와 같이 사용한다.

2.3 ICCP와 MMS와의 관계

(그림 2)는 ICCP와 MMS의 관계를 도식화한 것이다. ICCP는 MMS상에서 동작하며 MMS의 서비스와 프로토콜 그리고 기능을 이용하는 표준화된 응용프로토콜이다.



(그림 2) ICCP와 MMS와의 관계

이것은 MMS 객체에 매핑되어 있는 구조화된 데이터를 표현하고, 그것에 대해 특정 의미를 지정함으로써 MMS의 기능을 향상시킬 수 있다. 실제적인 MMS 서비스의 예로서, MMS는 원거리 시스템에서 데이터를 읽고 쓰는 것이 가능하며 이 데이터는 어떠한 조건 없이 응답한다. 만일 특정 조건 하에서 데이터를 읽어야 할 경우, ICCP는 MMS에서는 제공되지 않는 서비

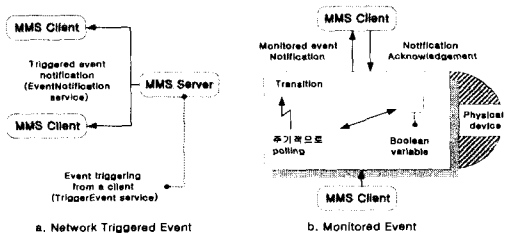
스를 제공한다. 이러한 것들을 수행하기 위해 ICCP는 제어센터의 요구조건을 충족시키는 CBB(Conformance Building Block)를 정의함으로써 최적화되고 효과적인 구현을 할 수 있다[3]. ICCP와 MMS는 둘 다 관련된 사용자에게 서비스를 제공한다는 점에서는 같지만, ICCP는 전력센터간에 메시지교환을 위해 최적화되었다는 점에서는 다르다.

### 3. MMS와 ICCP에서의 Event Management 비교

#### 3.1 MMS에서의 이벤트 관리(Event Management)

MMS 이벤트 관리는 MMS의 일부분으로서 서버에서 미리 정의된 이벤트가 발생했을 때 이것을 클라이언트에게 통보하며, 이러한 이벤트에 해당하는 행동을 취한다. MMS 서버는 클라이언트로부터 서비스 요청을 받을 때 이것을 받아 들여 처리할 뿐 아니라 이들 이벤트에 등록된 클라이언트에게 이벤트 통보(event notification)를 보내야 한다. MMS의 일부분인 이벤트 관리는 ISO(International Standard Organization)에 의해 다음과 같은 3가지 객체로 나뉜다.

- 이벤트 조건(Event Condition : EC) 객체  
이벤트가 발생했는지 탐지하는 것에 필요한 정보를 포함하고 있으며 해당 이벤트와 연관된 클라이언트를 나타낸다.
- 이벤트 행동(Event Action : EA) 객체  
이벤트가 발생했을 때 MMS 서비스의 실행에 관련된 상태 정보를 가지고 있다.
- 이벤트 등록(Event Enrollment : EE) 객체  
하나의 이벤트 조건과 하나의 이벤트 행동을 연관시켜 해당 클라이언트에게 이벤트 통보를 보내기 위한 정보를 포함하고 있다.



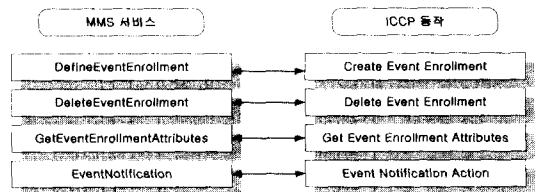
(그림 3) MMS에서 이벤트의 종류

MMS는 두 가지 형태의 이벤트를 가지고 있다.

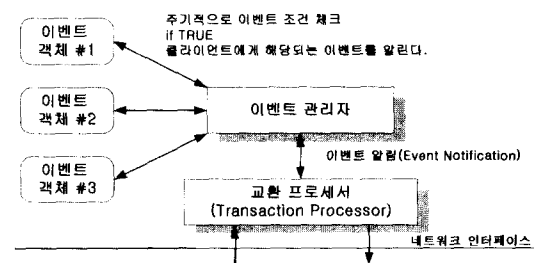
- Network-Triggered event  
MMS 클라이언트로부터 이벤트 요청을 받아들여 해당 이벤트를 처리한 후 이런 이벤트와 관련이 있는 클라이언트에게 통보한다.
- Monitored event  
MMS 서버는 해당 이벤트가 발생했는지를 검사하기 위해 주기적으로 부울 변수 값을 읽어들이고 이것을 해당하는 이벤트 조건의 상태(state) 값과 비교한다. 만일 차이가 발견되면, MMS 서버는 이벤트 전이 처리(Event Transition Processing : ETP)를 위한 절차를 실행시켜 발생된 이벤트에 대해 등록된 행동을 취한 후 해당 클라이언트에게 이벤트 발생을 알린다. 여기서 부울 변수 값은 서버에 의해 모델링된 물리적 디바이스의 상태를 나타내는 것이다.

#### 3.2 ICCP에서의 이벤트 관리

(그림 4)는 MMS와 ICCP의 이벤트에 대한 서비스의 매핑관계를 보여준다. (그림 5)는 ICCP에서 이벤트가 일어났는가를 체크하는 방법을 보여준다. ICCP에서는 이벤트가 일어났는지 주기적으로 감시한다. 이벤트가 발생했을 때, 교환 프로세서는 이벤트 관리자로부터 이벤트가 발생했음을 알려오면 이것을 처리할 사건 등록 객체를 참조하여 해당 클라이언트에게 알리는 역할 등을 한 후 이 객체는 제거된다. ICCP에서의 이벤트 객체에는 이벤트 등록 객체, 이벤트 조건 객체 두



(그림 4) MMS와 ICCP에서 이벤트 서비스 비교

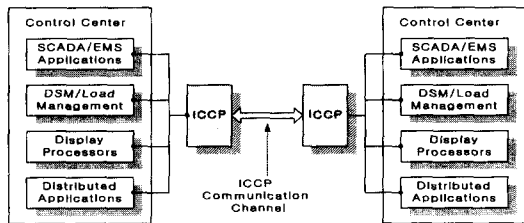


(그림 5) ICCP에서의 이벤트 모니터링

가지가 있다. 이벤트 등록 객체는 미리 정의된 이벤트 조건이 발생했을 경우, 행하여지는 동작에 관한 정보를 가지고 있다. 이러한 정보에는 통보를 필요로 하는 클라이언트의 이름과 통보 받기를 원하는 이벤트 조건의 이름이 포함된다. 이벤트 조건 객체는 이벤트가 일어나는 조건을 나타내는 객체로서 MMS와는 달리 이벤트 조건 객체는 동적으로 변화될 수 없고 가상제어센터 내에 정적으로 존재한다. ICCP의 이벤트 처리방식은 MMS와는 다른 방식으로 행하여지는데 ICCP에서는 이벤트 행동 객체를 지정하지 않고 사건의 발생만을 클라이언트에 통보하고, 그 이후의 처리는 전적으로 클라이언트의 몫이 된다.

#### 4. ICCP의 사용 모델

ICCP를 사용하면 전력센터내의 EMS/SCADA, 전기 수요관리(Demand-Side Management : DSM), 부하관리(load management) 그리고 분산된 응용(distributed application) 등 서로 다른 여러 프로세서간의 통신을 이룰 수 있다[3]. 각 시스템간의 통신시 각각의 통신은 통신 규약을 따르는 대신 (그림 6)과 같이 ICCP 하나의 통신 채널을 이용하여 통신할 수 있다. 그러므로 시스템 통합을 할 때 각각의 통신규약을 이용하는 것보다 적은 비용으로 통신을 할 수 있고, 보다 편리한 통합을 이룰 수 있다.



(그림 6) ICCP를 이용한 전력센터간의 통신모델

##### 4.1 가상제어센터(VCC)

제어센터는 하나 또는 그 이상의 가상제어센터로 모델링되며 가상제어센터는 MMS 가상제조기로 매핑된다. 따라서 가상제어센터는 MMS 가상제조기와 같은 속성과 동작을 가지고 있다. MMS 가상제조기는 제조환경의 실제 디바이스의 기능을 매핑한 것으로 서버기능의 집결체이다. MMS 가상제조기는 응용프로그램과 실제 디바이스 사이에 위치하여 기계의 실제

기능과 동작을 MMS의 추상적인 객체로 매핑하는 수단을 제공한다. ICCP 가상제어센터를 MMS 가상제조기로 매핑함으로써 ICCP 가상제어센터는 MMS 가상제조기가 실제 디바이스를 위해 행하는 것처럼 제어센터를 위해 같은 기능을 수행한다. 가상제어센터의 의미와 관련지어 ICCP에서는 클라이언트와 서버를 정의한다. 서버는 특별한 서비스 요청에 대해 가상제어센터로서 행동하는 통신 개체(communication entity)로 볼 수 있다. 클라이언트는 ICCP 서비스 요청을 위해서 가상제어센터를 이용하는 통신 개체로 정의할 수 있다.

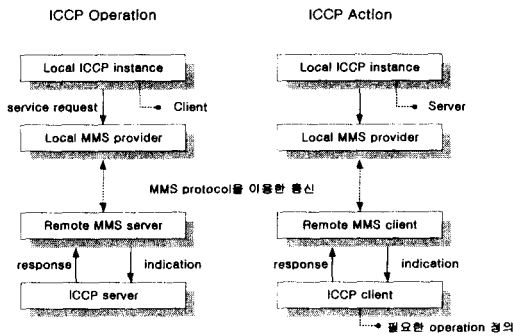
##### 4.2 ICCP 동작

MMS와 ICCP에서 도메인(domain)은 가상제조기 혹은 가상제어센터의 기능을 표현한 것으로 가상제조기의 부분집합이며 실제 기기가 보유한 물리적/논리적 자원을 나타낸다. 공장자동화 과정에서 많이 쓰이는 로봇의 예를 들어보자. 로봇의 팔은 로봇의 일부분으로 도메인이라고 할 수 있을 것이다. 이 때 단지 로봇의 팔 자체만을 의미하는 것이 아니라 로봇 팔의 제어프로그램을 위한 데이터와 코드를 포함하여 실제 로봇 팔의 동작을 위해 필요한 모든 자원들을 도메인이라고 한다. 도메인은 다음과 같은 두 가지 요소로 구성되어 있다.

- Capability의 집합 : Capability는 실제 디바이스가 가지고 있는 모든 자원을 의미.
- Information : 프로그램 실행을 위해 사용되는 코드와 데이터 영역을 나타낸다.

ICCP는 논리적인 개체로서 동작하는 하나 또는 그 이상의 프로세스로서 모델링된다. 여기서 논리적인 개체라는 것은 제어센터가 데이터를 얻고, 교환할 때 사용하는 통신을 제공하는 것을 뜻한다. 이러한 프로세스 사이에 통신을 하기 위해 서비스와 프로토콜을 정의하고 있다. ICCP 명세서에는 많은 동작과 행동을 정의하고 있다. (그림 7)은 ICCP에서의 동작과 행동에 대한 그림이다. 그림의 왼쪽은 동작 과정을 나타내며, 오른쪽은 행동 과정을 나타낸다. ICCP 동작은 ICCP 클라이언트와, ICCP 행동은 ICCP 서버와 관련된다. ICCP 클라이언트로 동작하는 지역내의 ICCP 인스턴스에서 서비스를 요청할 때 이것은 같은 지역내의 MMS 제공자에게 전달되며, MMS 제공자는 MMS 프로토콜을 이용하여 원거리의 MMS 통신을 시작한다. 원거리

의 MMS 서버는 ICCP 클라이언트로부터의 서비스 요청을 ICCP 서버에게 알리며(indication), 이것에 대해 ICCP 서버는 MMS 서버에게 응답한다(response). ICCP 행동은 ICCP 서버로부터 시작된다. ICCP 서버에서 타 이머가 종료되었다든지 하는 자체 이벤트에 대해 ICCP 클라이언트에게 보고한다.

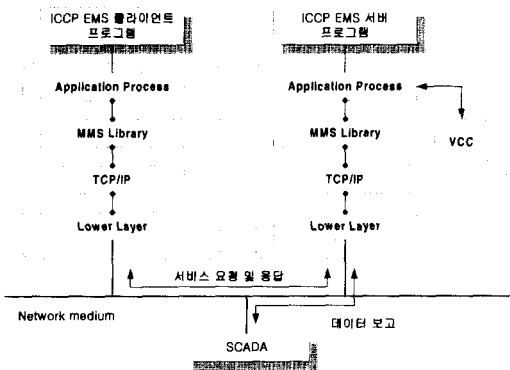


(그림 7) ICCP의 동작과정

### 5. ICCP 구현

#### 5.1 전체 시스템 구성 환경

본 논문에서는 OSI 모델에 비해 일반적인 TCP/IP 를 ICCP 구현을 위해 선택하였다.



(그림 8) 프로그램 구성 환경

ICCP는 여러 EMS에 사용될 수 있으며 또한 EMS 와 SCADA간의 데이터 교환에 사용될 수 있다[8]. 본 구현에서는 두 가지 형태의 데이터 교환을 모델링하였다. 서버 EMS에 데이터를 요구하는 EMS 클라이언트 프로그램과 클라이언트의 요구에 응답하는 EMS 서버

프로그램, 그리고 EMS 서버 프로그램에 데이터를 전달하는 SCADA 프로그램으로 구성하였다. (그림 8)은 본 논문에서 구현한 ICCP의 전체 환경을 나타낸 것이다.

EMS 클라이언트로부터의 서비스 요청은 응용프로세스(Application Process : AP)에서 요청한 서비스에 해당하는 PDU(Protocol Data Unit)를 MMS 라이브러리(library)로 구성하고, 네트워크를 통해 EMS 서버로 전달된다. 서버 쪽의 응용프로세스에서는 요청 받은 서비스에 대해 클라이언트에서와 같은 방식으로 MMS 라이브러리를 이용하여 응답을 보내게 된다.

#### 5.2 연결설정을 위한 MMS 라이브러리 호출

ICCP 명세서에는 구현단계별로 컨포먼스 블록(conformance block)으로 나누어 정의하고 있다. 본 논문에서는 전체 9개의 블록 가운데 가장 핵심적인 부분인 블록 I 즉, 연결설정 객체에 관련된 서비스와 데이터 값(data value)과 데이터 집합(data set) 객체에 관련된 서비스를 구현하였다. <표 1>은 본 논문에서 구현된 컨포먼스 블록과 그에 해당하는 객체와 서비스를 나타낸 것이다.

<표 1> 컨포먼스 블록과 구현 범위

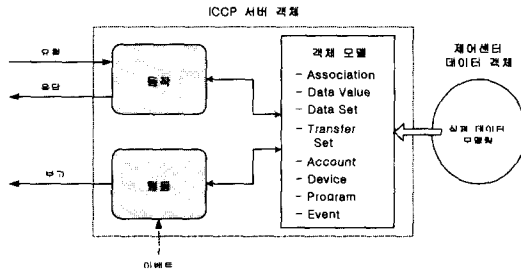
컨포먼스 블록	구현객체와 서비스
블록 I	1. 연결(association) 2. 데이터 값 3. 데이터 집합 관련 서비스 4. 전송집합 일부

ICCP는 MMS를 하부 계층 규약으로 지정하고 있다. ICCP의 서비스 함수들이 MMS의 서비스 함수들을 호출하여 사용하는 것이므로 ICCP를 구현하기 위해서는 MMS 라이브러리가 필요한데 본 논문에서는 MMS 라이브러리 86개의 서비스 중 일부분을 이용하였다. MMS 라이브러리를 사용하기 위해서는 AR Name(Application Reference Name)을 지정하여 통신을 한다. 이 AR Name은 응용프로그램이 있는 컴퓨터의 주소와 프레젠테이션 주소 등을 가지고 있다. 따라서 사용자는 MS Access 또는 SISCO Stack Configuration Utility를 이용하여 AR Name, 라우팅(routing) 그리고 현재 사용하고 있는 네트워크에 대한 정보들을 입력할 수 있다[3, 4]. Stack Configuration Utility를 사용하여 AR Name을 지정한 후 사용자 프로그램에서는 정의된 AR Name을 사용하여야 한다.

5.3 컴포넌스 블록 I

ICCP는 실제 전력센터를 모델링하기 위해서 10개의 객체를 사용한다. 모든 객체들은 MMS의 객체들로 매핑이 되는데 ICCP의 객체들로는 Association, Data Value, Data Set, Account, Information Message, Transfer Set, Device, Program, 그리고 Event Enrollment와 Event Condition이 있다[3]. 이와 같은 객체들에 접근하기 위한 동작과 행동이 정의되는데, 정의된 모든 동작과 행동들 또한 MMS의 서비스들로 매핑된다.

ICCP는 실제 데이터 센터를 나타내기 위해서 가상 제어센터라는 추상적인 개념을 사용한다. 이는 MMS의 가상제조기기 개념을 확장시킨 것이며 실제 전력센터의 모든 데이터들을 추상화하여 나타낸다. (그림 9)와 같이 제어센터에 존재하는 실제 데이터들을 객체화하여 가상제어센터 내에 나타내어 다른 제어센터와의 통신에 이용된다.



(그림 9) ICCP 객체 모델

5.3.1 연결 객체(association object)

연결은 원격지 제어센터와의 연결설정 및 해제에 관계되는 객체이다. 연결식별자(association identifier)는 ICCP간의 연결을 특정하게 나타내는 것이고, MMS의 연결식별자로 매핑된다. 연결참조(association reference)는 ICCP 클라이언트의 연결 프로세스와 연결 개체를 나타낸다[5, 9]. QoS(Quality of Service)는 연결의 다양한 서비스의 질을 나타낸다. 서비스로는 연결을 요청하는 Initiate, 연결을 종료하는 Conclude 그리고 Abort 서비스가 있다.

5.3.2 데이터 값 객체

데이터 값 객체는 원격 제어센터 내에 설정된 변수들을 나타낸다. 예를 들면 전류 혹은 전압과 같은 실제 데이터를 변수로 설정할 수 있으며, 이러한 변수는 ICCP에서는 데이터 값 객체로 추상화하여 표현하고 있다. 이 데이터 값 객체는 실제적으로 가장 많이 쓰이

는 객체이다. 데이터 값 객체는 MMS 변수 객체에 매핑된다. 데이터 값 객체는 VMD-Specific 또는 Domain-Specific의 범위를 갖는다. 데이터 값 객체가 VMD-Specific으로 설정하여 구현하였을 경우 같은 가상제어센터 내에서 참조하여 사용할 수 있음을 의미하고, Domain-Specific으로 설정하여 구현하였을 경우에는 특정 도메인 내에서만 참조되고 접근할 수 있음을 의미한다. 모든 데이터 값 객체는 MMS 타입을 갖는 변수로 매핑되는데, 여기서 타입은 변수가 가질 수 있는 값의 범위를 의미한다. 정수형 실수형 혹은 1 옥텟(octet) 등의 범위를 의미한다. 서비스로는 데이터 값 객체로부터 값을 얻어내는 GetDataValue, 데이터 값 객체에 값을 지정하는 SetDataValue, 데이터 값 객체의 이름을 얻어내는 GetDataValueNames 그리고 데이터 값 객체의 타입을 얻어내는 GetDataValueType이 있다. 이들 서비스들은 MMS의 Read, Write, GetVariableAccessAttribute 그리고 GetNameList 등으로 매핑되어서 사용된다.

(그림 10)은 ICCP의 데이터 값 객체 접근을 위해 사용된 MMS의 Read 서비스의 ASN.1 표현의 일부이며, (그림 11)은 ASN.1 표현을 C언어의 구조체로 바꾼 것이다.

```

Read-Request ::= SEQUENCE {
    specificationWithResult [0] IMPLICIT BOOLEAN DEFAULT FALSE,
    variableAccessSpecification [1] VariableAccessSpecification
}
VariableAccessSpecification ::= CHOICE {
    listOfVariable [0] IMPLICIT SEQUENCE OF SEQUENCE {
        variableSpecification VariableSpecification,
        alterAccess [5] IMPLICIT AlternateAccess OPTIONAL
    },
    variableListName [1] ObjectName
}
    
```

(그림 10) ICCP 데이터 값 객체 접근 위해 사용된 MMS Read 서비스의 ASN.1 표현

```

typedef struct Read_Request /* SEQUENCE */
{
    AsnBool* specificationWithResult; /* IMPLICIT BOOLEAN DEFAULT FALSE */
    struct VariableAccessSpecification* variableAccessSpecification;
} Read_Request;
typedef struct VariableAccessSpecification /* CHOICE */
{
    enum VariableAccessSpecificationChoiceId
    {
        VARIABLEACCESSSPECIFICATION_LISTOFVARIABLE,
        VARIABLEACCESSSPECIFICATION_VARIABLELISTNAME
    }
    union VariableAccessSpecificationChoiceUnion
    {
        VariableAccessSpecificationSeqOf listOfVariable;
        struct ObjectName* variableListName; /* [1] ObjectName */
    };
} VariableAccessSpecification;
    
```

(그림 11) ICCP 데이터 값 객체 접근을 위해 사용된 MMS Read 서비스의 C언어 표현

5.3.3 데이터 집합 객체

데이터 집합 객체는 ICCP 사용자에게 의해서 데이터 값 객체들을 그룹화하는 것을 나타낸다. 그러므로 좀 더 편리한 데이터 접근을 지원한다. 단 한 번의 요청으로 하나 이상의 데이터 값 객체에 대해 동작을 취할 수 있게 된다. 데이터 집합 객체는 다음과 같은 형태로 정의된다.

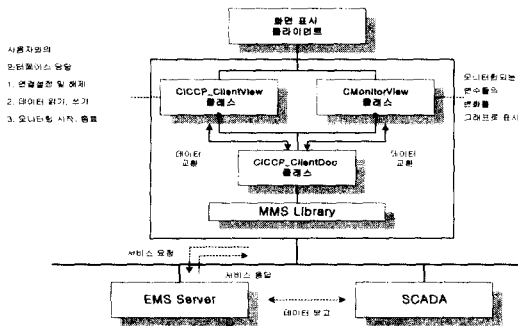
```

Model : Data Set
Key Attribute : Data Set Name
Attribute : Data Set Scope(VCC-specific,
            ICC-specific)
Attribute : Transfer Set Name
Attribute : DSConditions Detected
Attribute : Event Code Detected
Attribute : Transfer Set Time Stamp
Attribute : List of Data Objects
Attribute : List of Access Specification
    
```

관련되는 동작으로는 데이터 집합 객체를 만드는 CreateDataSet, 이미 존재하는 데이터 집합 객체를 없애는 DeleteDataSet, 데이터 집합 객체로부터 연결된 구조를 가지는 값들을 한꺼번에 얻어내는 GetDataSetElementValues, 데이터 집합 객체에 값들을 지정하는 SetDataSetElementValues, 데이터 집합 객체의 이름을 얻어내는 GetDataSetNames, 그리고 데이터 집합 객체 내의 하나 이상의 변수 이름을 얻어내는 GetDataSetElementNames가 있다.

5.4 응용프로그램

응용프로그램은 Visual C++ 5.0을 이용해 구현하였다. 화면 표시 클라이언트를 포함한 EMS 클라이언트는 윈도우 환경에서 동작하도록 구현하였다. EMS 서버와 SCADA는 콘솔에서 실행되도록 구현하였다. ICCP의 하부에 위치한 MMS 라이브러리는 ANSI-C로 구



(그림 12) 응용프로그램의 전체 구조

현하여 함수 호출을 통해 ICCP와 매핑하였다. (그림 12)는 응용프로그램의 전체 구조를 나타낸다.

<표 2>는 EMS 클라이언트, EMS 서버와 SCADA에 구현 적용한 ICCP 서비스를 나타냈다.

<표 2> 구현한 ICCP 서비스

프로그램	관련 객체	구현된 ICCP 서비스
EMS 클라이언트	연결 설정	Initiate, Conclude, Abort
	가상제어센터	GetNameList, Identify
	데이터 값	GetDataValue, SetDataValue, GetDataValueAttributes
EMS 서버	데이터 값	CreateDataSet, DeleteDataSet, GetDataSetElementValues, GetDataSetAttributes
	데이터 집합	GetDataValue, SetDataValue, GetDataValueAttributes
	데이터 집합	CreateDataSet, DeleteDataSet, GetDataSetElementValues, GetDataSetAttributes
SCADA	연결 설정	Initiate, Conclude
	가상제어센터	GetNameList, Identify
	데이터 값	GetDataValue, SetDataValue, InformationReport

SCADA 프로그램은 각 모선의 데이터들 즉, 전압, 전류, 역률, 주파수 등의 값을 생성하여 Transfer-Report 서비스를 이용하여 주기적으로 EMS 서버에게 전달한다.

EMS 서버 프로그램은 ICCP의 컨포먼스 블록 I에 해당하는 기본 서비스의 알림과 응답의 부분을 갖고 있으며 SCADA 시스템에서 생성한 모선 데이터들을 보고 받아서 EMS 클라이언트 프로그램의 서비스 요구에 응답한다.

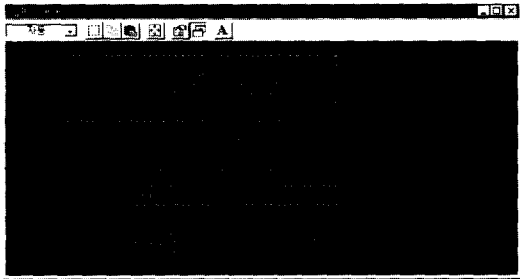
EMS 클라이언트 프로그램은 분할윈도우를 사용한 GUI(Graphic User Interface)로 구현하였다. 하나의 윈도우는 ICCP의 컨포먼스 블록 I의 기본서비스 즉, 연결설정과 기본 데이터에 대한 읽기, 쓰기의 요청 부분과 감시의 시작과 종료에 대한 것을 사용자가 입력할 수 있도록 대화상자를 구성하였고 서버에 존재하는 여러 데이터들을 감시할 수 있는 화면을 구성하였다.

5.4.1 EMS 서버 프로그램

EMS 서버 프로그램은 MFC(Microsoft Foundation Class)의 콘솔 응용(console application)으로 만들어졌다. 하부 계층 SCADA 프로그램과 연결을 설정하고 버스 데이터를 보고 받는다. 그리고 EMS 클라이언트



의 요청에 응답을 한다. 서비스의 인식과 응답부분이 구현되었으며 SCADA 시스템과 연결을 설정하고 데이터를 받아들이는 부분이 첨가되었다.

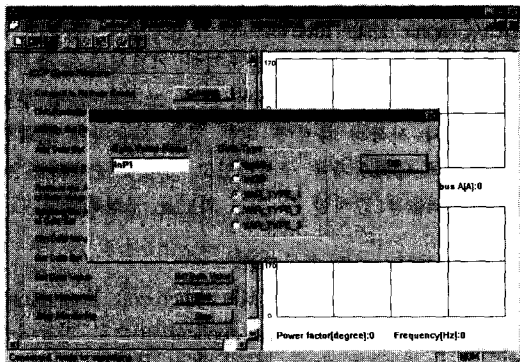


(그림 13) EMS 서버 프로그램의 실행 화면

사용된 변수로는 SCADA 시스템으로부터 보고 받는 모선 A, B, C의 전압, 전류, 역률, 주파수, SCADA 시스템으로부터 보고 받은 데이터로부터 계산된 모선 A, B, C의 무효전력과 유효전력 그리고, 모선 A, B, C의 각각에 대한 전류, 전압, 역률 등의 현재 값을 읽을 수 있도록 설정한 IndicationPoint형[5] InP1, InP2, InP3가 있다.

5.4.2 EMS 클라이언트 프로그램

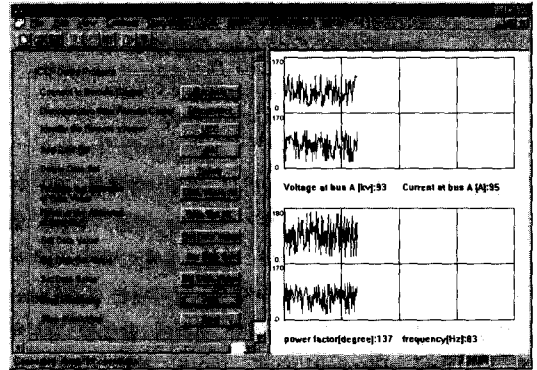
EMS 클라이언트 프로그램은 MFC를 사용한 Win32 응용프로그램으로 만들어졌다. ICCP에서 제공하는 기본서비스와 이런 서비스를 바탕으로 하여 응용서비스를 구현하였다. 사용자가 Start Monitoring을 클릭할 때 연결된 EMS 서버로부터 전압, 전류, 역률, 주파수 값들을 얻어내게 되며 이러한 실시간 데이터는 사용자에게 보여지게 된다.



(그림 14) 데이터 값을 얻어내는 대화 화면

사용자의 입력을 받아 들여서 서비스를 실행하는 것과 서버에서 들어오는 데이터들을 받아서 표시하는 두 개의 일을 동시에 처리하기 위해서 멀티 쓰레드를 사용하여 구현하였다. 이는 서버에서 발생한 이벤트에 대한 데이터를 받을 때 새로운 함수를 실행하도록 했다.

EMS 클라이언트 프로그램은 사용자의 입력을 처리하기 위한 대화상자와 모니터링되는 변수들을 그래프로 나타내주는 부분으로 구성되어져 있다.



(그림 15) ICCP 데이터 모니터링 화면

5.4.3 SCADA 프로그램

SCADA 프로그램은 Visual C++5.0의 콘솔 응용을 사용한 도스용 응용프로그램이다. 각 모선의 데이터들을 생성하고 그것들을 EMS 서버 프로그램에 전달한다. 데이터들은 모선 A, B, C의 전압, 전류, 역률 그리고 주파수이며 이러한 데이터들은 InformationReport 서비스를 이용하여 EMS에 전달된다. 프로그램의 변수는 모선 A, B, C의 전압, 전류, 역률, 주파수, 총 12가지의 데이터이며 이를 EMS 서버에게 보고한다.

이상과 같이 본 논문에서 구현된 프로그램을 모듈별로 나누어 살펴보았다. EMS 클라이언트 시스템과 EMS 서버 시스템간에 교환되는 데이터는 SCADA 시스템 데이터인 IndicationPoint 형의 데이터, ControlPoint 형의 데이터, TA\_SEGMENT\_PROFILE 형태의 데이터 그리고 각 모선의 전압, 전류, 역률, 주파수와 무효전력, 유효전력 그리고 온도 등과 같은 데이터들이 사용되었다. 위와 같은 세 개의 프로그램으로 EMS 시스템간, EMS 시스템과 SCADA 시스템간의 ICCP를 사용한 통신을 구현하였다.

## 6. 결 론

현재의 전력시스템은 EMS/SCADA의 형태로 구성되어 많은 양의 데이터를 효과적으로 처리하고 있다. 하지만 각각의 시스템들은 업체에 따라서 다양한 형태의 구조와 프로그래밍 언어를 가지므로 모든 시스템간의 통신을 지원하는 데에는 어려움이 많다. 이에 따라 대두된 것이 시스템에 독립적인 통신규약으로 전력분야에서는 ICCP(Inter-Control Center Protocol)가 제안되었다. ICCP는 각 제어센터간에 실시간 데이터, 제어 데이터, 측정 데이터를 정해진 규약에 따라 일정한 포맷으로 전송하는 프로토콜로 확장성이 뛰어난 장점을 가지고 있다. 본 논문에서는 ICCP를 구현하기 위해 표준 통신규약인 MMS를 전력센터간의 통신에 응용하였다. 자동화 네트워크의 프로토콜은 원래 OSI 7 계층 참조 모델 하에서 규정되고 있지만 실시간 제어에 보

다 유리한 TCP/IP 하에서의 구현으로 나아가고 있는 추세이다. 따라서 본 논문에서도 TCP/IP를 기반으로 하여 ICCP에 정의된 기본적인 핵심기능과 이것을 간단한 모델에 적용하여 이 프로토콜의 동작과 기능을 구현하였다. 그럼으로써 전력센터내의 존재하는 여러 시스템간의 통신에서 발생할 수 있는 서로 다른 기종, 업체간의 통신의 문제점을 해결할 수 있는 통신규약인 ICCP를 새로운 통신 방식으로 제시하였다. 또한 ICCP와 MMS 서비스의 매핑관계를 분석하고, 특히 ICCP와 MMS에서의 이벤트 관리를 분석하였다. ICCP가 어떠한 방식으로 MMS를 사용하는가를 MMS 라이브러리를 사용하여 확인하였으며, EMS 시스템간의 통신과 EMS와 SCADA 시스템간의 통신을 TCP/IP 하에서 윈도우 기반의 프로그램으로 구현하여 보다 편리한 환경을 제공하였다.

## 부 록

〈표 A.1〉 EMS 클라이언트에서 쓰인 주요 함수

	함수	내용
CICCP_Client Doc	void CICCP_ClientDoc ::OnConnect()	연결을 시도하는 함수이다. Initiate 서비스에 필요한 정보를 구성한 후 전송한다.
	void CICCP_ClientDoc ::OnDisconnect()	연결중인 센터와의 연결을 끝내는 함수이다.
	void CICCP_ClientDoc ::OnIdentify()	연결되어 있는 센터에 관한 정보를 얻어내는 함수이다.
	void CICCP_ClientDoc ::GetName()	연결되어 있는 센터에 존재하는 Data Value와 Data Set의 이름을 얻어내는 함수이다.
	void CICCP_ClientDoc ::Addset()	연결되어 있는 센터에 Data Set을 설정하는 함수이다.
	void CICCP_ClientDoc ::Deleteset()	연결되어 있는 서버의 설정된 Data Set을 없애는 함수이다.
	void CICCP_ClientDoc ::OnGetDataValueAtt()	서버에 설정된 Data Value의 특성을 얻어내는 함수이다.
	void CICCP_ClientDoc ::OnGetDataSetAtt()	서버에 설정된 Data Set의 구성을 얻어내는 함수이다.
	void CICCP_ClientDoc ::OnGetDataValue()	서버에 존재하는 Data Value의 값을 얻어내는 함수이다. 데이터의 값을 얻는 때는 변수의 타입을 지정하여야 한다. 변수의 타입은 CICCP_ClientApp 클래스에 멤버 변수로 정의되어있다.
	void CICCP_ClientDoc ::OnGetDataSetValue()	서버에 설정된 데이터 집합의 값을 얻어내는 함수이다. 데이터 집합은 이름으로 식별되며 데이터 집합에 지정된 데이터 값들의 값을 얻어낸다.
	void CICCP_ClientDoc ::OnSetDataValue()	서버에 설정된 데이터 값의 값을 지정하는 함수이다. 변수들의 이름, 타입 그리고 원하는 값을 입력하여 데이터의 값을 바꿀 수 있다.
void CICCP_ClientDoc ::OnStartMonitoring()	ICCP 응용서비스를 시작하는 함수이다. Set Data Value서비스를 이용하여서 서버에 지정된 변수의 값을 100으로 지정해두면 서버는 버스의 데이터, 즉 전압, 진류, 역률 그리고 주파수를 Transfer Report 서비스를 이용하여 클라이언트에게 보고를 한다. 데이터들은 CMonitorView에서 받아들여져서 화면에 그려진다.	

<표 A-2> EMS 클라이언트에서 쓰인 주요 함수 (1)

	함 수	내 용
CICCP_Client Doc	void CICCP_ClientDoc ::OnStopMonitoring()	OnStartMonitoring에 의해서 시작된 보고를 Set Data Value 서비스를 이용하여 서버에 지정된 변수를 200으로 지정하여 데이터 보고를 정지시키는 함수이다.
	void CICCP_ClientDoc ::GetTaccount()	서버에 있는 Transfer Account 변수의 값을 얻어내는 함수이다. TAccount_Type이라는 타입의 변수를 Get Data Value 서비스를 이용하여 얻어내는 함수이다.
CICCP_Client View	void CICCP_ClientView ::OnFormConnect()	CICCP_ClientDoc::OnConnect()를 참조한다.
	void CICCP_ClientView ::OnFormDisconn()	CICCP_ClientDoc::OnDisconnect()를 참조한다.
	void CICCP_ClientView ::OnFormVcc()	CICCP_ClientDoc::OnIdentify와 CICCP_ClientDoc::OnGetname을 선택적으로 참조한다.
	void CICCP_ClientView ::OnFormAdd()	CICCP_ClientDoc::OnAddset()을 참조한다.
	void CICCP_ClientView ::OnFormDelete()	CICCP_ClientDoc::OnDeleteset()을 참조한다.
	void CICCP_ClientView ::OnFormAttValue()	CICCP_ClientDoc::OnGetDataValueAtt()를 참조한다.
	void CICCP_ClientView ::OnFormAttSet()	CICCP_ClientDoc::OnGetDataSetAtt()를 참조한다.
	void CICCP_ClientView ::OnFormGetDataValue()	CICCP_ClientDoc::OnGetDataValue()를 참조한다.
	void CICCP_ClientView ::OnFormGetDataSetValue()	CICCP_ClientDoc::OnGetDataSetValue()를 참조한다.
	void CICCP_ClientView ::OnFormSetDataValue()	CICCP_ClientDoc::OnSetDataValue()를 참조한다.
	void CICCP_ClientView ::OnFormStart()	CICCP_ClientDoc::OnStartMonitoring()를 참조한다.
	void CICCP_ClientView ::OnFormStop()	CICCP_ClientDoc::OnStopMonitoring()를 참조한다.

<표 A.3> EMS 클라이언트에서 쓰인 주요 함수 (2)

	함 수	내 용
CICCP_CMonitorView	void CMonitorView ::DrawBox(CDC*, CRect*)	화면에 표시될 그래프의 라인을 그리는 함수.
	void CMonitorView ::Drawline(CDC*, CRect*,int)	실제데이터를 화면에 표시하는 함수. 마지막인자는 현재 그려지는 위치를 나타낸다.
	void CMonitorView ::DrawIndex(CDC*)	그래프의 표식을 나타내어주는 함수.
	void CMonitorView ::WriteText(CDC*, CRect*,int)	그래프와 동시에 나타내어지는 변수 값에 대한 정보.

<표 A.4> SCADA에서 쓰인 주요 함수

	함 수	내 용
SCADA	int init_EMS_SCADA(void)	사용할 채널의 개수, 연결에 사용되는 프로토콜 데이터의 최대 길이, 응용프로그램을 나타내는 AR Name 그리고 사용할 변수, 타입등 SCADA 프로그램을 초기화한다.
	void sendDataToEMS(void)	모션 A, B, C의 데이터들을 EMS 서버 시스템에 보고한다.
	void GetData(void)	모션 A, B, C의 데이터들의 값을 생성한다.

〈표 A.5〉 EMS 서버에서 쓰인 주요 함수

	함수	내용
Server	int initICCP(void)	사용할 채널의 개수, 연결에 사용되는 프로토콜 데이터의 최대 길이, 응용프로그램을 나타내는 AR Name 등 프로그램을 초기화한다.
	int connectToSCADA(void)	데이터를 받아들이는 SCADA 시스템과의 연결을 하는 함수이다.
	int Conclude(void)	SCADA 시스템과의 연결을 종료하는 함수.
	void ADD_TYPE(void)	데이터 교환에 사용될 타입을 설정하는 함수
	void ADD_VARIABLE(void)	데이터 교환에 사용될 변수를 설정하는 함수
	void setLocaldata(void)	설정된 데이터의 값을 얻어내는 함수. random() 함수를 사용하여 임의의 값을 발생한다.
	void SendInfo_data(void)	SCADA로부터 받은 데이터중 일부분의 데이터를 클라이언트의 요청에 의하여 보고하는 함수.
	void ReportService(void)	서버의 요청에 의해서 지정된 데이터들을 보고하는 함수 check_state()함수로 확인을 한 후 SendInfo_data()를 호출한다.

참 고 문 헌

[1] M. Brilland U. Gramm, "MMS : MAP Application Services for the Manufacturing Industry," Computer Networks and ISDN Vol.21, pp.357-380, 1991.

[2] P. Castori and P. Pleinvaux, "A Generic Architecture for MMS Servers," IEEE Transactions on Industrial Electronics, pp.211-218, June 1995.

[3] ICCP Inter-Control Center Communication Protocol ; Section 870-6-503 : Services and Protocol Version 6.2, Feb. 1997.

[4] ISO/IEC 9506-1 Industrial Automation Systems-Manufacturing Message Specification, Part 1, 1990.

[5] ICCP Inter-Control Center Communication Protocol; Section 870-6-802 : Object Model, Version 6.2, Feb. 1997.

[6] ISO/IEC 9506-2 Industrial Automation Systems-Manufacturing Message Specification, Part 2, 1990.

[7] MMS-EASE reference Manual, Revision 11, SISCO Inc, 1997.

[8] ICCP User Guide, EPRI Project Manger, Final Draft, 1996.

[9] S. Carullo et al, "Integrating a Power Systems Laboratory into a Client/Server based Computing Environment," Proc. of Amer. Soc. for Eng. 1996.

[10] ISO 8824, Information processing systems-Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1), 1995.

[11] ISO 8825, Information processing systems-Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1), 1987.

[12] Peter A. Lagoni, Christophei, Crall, and Thomas G. Bartz, "HP MAP 3.0 Manufacturing Message Specification/800," Hewlett-Packard Journal, Aug. 1990.

[13] Raymond Seng-Sim Cheah *et al*, "Design and implementation of an MMS environment on ISODE", Elsevier Science B.V. Computer Communications 20, pp.1354-1364, 1997.

[14] A. Valenzano, C. Dermartini and L. Ciminiera. "MAP and TOP Communications," Addison-Wesley, Massachusetts, 1992.

[15] Jed Caben and John Jackman, "An Icon-Based Approach to System Control Development," IEEE Trans. on Industrial Electronics, Vol.37, No.3, pp. 259-264, Jun. 1990.

[16] E. Freund, H.-J. Buxbaum and U. van der Valk, "PC-based hierarchical manufacturing cell control", Control Eng. Practice, Vol.1, No.6, pp.1047-1054, 1993.



장 경 수

e-mail : kschang@ece.skku.ac.kr  
 1994년 성균관대학교 전기공학과 졸업(학사)  
 1998년 성균관대학교 대학원 (공학석사)  
 2000년 성균관대학교 대학원 전기 전자 및 컴퓨터공학과 박사과정 수료

1994년~1995년 LG산전(주)  
 관심분야 : Industrial Networks, Wired/Wireless ATM, Discrete Event Systems 등



**장 병 옥**

e-mail : bwchang@samsung.com  
1998년 성균관대학교 제어계측공  
학과(학사)  
2000년 성균관대학교 전기전자 및  
컴퓨터공학과 대학원  
(공학석사)

2000년~현재 삼성전자(주) 이동통신개발팀  
관심분야 : Industrial Networks, Network Management 등



**한 경 덕**

e-mail : kdhahn@ece.skku.ac.kr  
1997년 성균관대학교 전기공학과  
(학사)  
1999년 성균관대학교 전기전자 및  
컴퓨터공학과 대학원  
(공학석사)

1999년~2000년 한솔 텔레콤(주)  
2000년~현재 인포섹코리아(주)  
관심분야 : Industrial Networks, Network Management 등



**신 동 렬**

e-mail : drshin@ece.skku.ac.kr  
1980년 성균관대 졸업(학사)  
1982년 한국과학기술원  
(공학석사)  
1992년 Georgia Institute of Tech.  
(Ph.D)

1994년~현재 성균관대학교 전기전자 및 컴퓨터공학부  
부교수  
1982년~1986년 Assistant researcher, Daewoo Heavy  
Ind., Limited  
1992년~1994년 Senior researcher, Samsung Data  
Systems  
관심분야 : Industrial networks including MMS,  
ICCP, and Fieldbus. High-speed telecom-  
munication networks with emphasis on  
ATM modelling and switching. Perform-  
ance evaluation with markovian and  
queueing networks 등