멀티미디어 회의 시스템을 위한 다중점 통신 서비스의 구현

성 동 수[†] · 현 동 환[††] · 함 진 호[†††]

요 약

현재 및 미래의 다양한 통신망에서 멀티미디어 회의 관련 응용 서비스를 구현하기 위해서는 기본적인 회의 서비스에 관련된 규약이 필요하며 이와 관련하여 제정된 규약이 MCS(Multipoint Communication Service)이다.

본 논문은 멀티미디어 회의 시스템을 위하여 국제 표준으로 권고되어 있는 다지점 통신 서비스(MCS)를 구현하기 위하여 관련 자료를 분석하고, 이를 근간으로하여 알고리즘의 제안하고 이를 UNIX machine 및 windows/NT machine에서 구현하였다.

Implementation of Multipoint Communication Service for Multimedia Conference System

Dong Su Seong[†] · Dong Hwan Hyun[††] · Jin Ho Hahm[†††]

ABSTRACT

For the implementation of a multimedia conference applications in a various communication network, the fundamental conference services are needed. MCS(Multipoint Communication Services) is recommended for this purpose.

In this paper, the related papers are analyzed for the implementation of MCS that is recommended in international standard for a multimedia conference. Based on this, the algorithm is proposed and implemented in unix machine and Windows/NT machine.

1. 서 론

현재 및 미래의 다양한 통신망에서 멀티미디어 회의 관련 응용 서비스를 구현하기 위해서는 기본적인 회의 서비스에 관련된 규약이 필요하며[1-4] 이와 관련하여 제정된 규약이 MCS(Multipoint Communication Service)[5][6]이다. MCS는 상호대화적인 다양한 멀티미디어 응용들을 지원하기 위하여 설계되었으며, T.123[7]에서 규정한 다양한 Network상에서 다양한 여러 응용 객체들 사이에 양방향 다자간 통신을 제공한다. 또한 T.125는 MCS 프로토콜 규격으로, MCS provider들 사이의 데이타 및 제어 정보의 교환을 위한 규약을 위한 과정들을 소개하고, 데이타 및 제어 정보의 교환에 사용되는 MCS PDU(Protocol Data Unit)들의 구조 및 부호화를 정의하고 있다[6]. 즉, T.122는 사용자가 MCS를 실제적으로 이용하기 위한 MCS Primitive들을 소개하고 있으며, T.125는 T.122에서 소개한 MCS Primitive들을 실제적으로 구현하

※본 연구결과는 정보통신부 정책과제인 정보통신 표준화 사업의 일환으로 수행됨

† 정 회 원:경기대학교 공과대학 전자공학과
†† 정 회 원:영진 전문대학 사무자동화학과
††† 정 회 원:한국전자통신연구원 멀티미디어 표준 연구실
논문접수:1996년 5월 8일, 심사완료:1997년 4월 15일

기 위한 방안을 소개하고 있다[5]. MCS는 일대일의 MCS연결들을 통하여 다지점 영역을 형성한다. 이 다지점 영역 내에서 임의의 응용 객체는 영역내 다른 객체들에게 채널들을 이용하여 정보를 전달할 수 있으며, 자원 충돌 해결을 위하여 토큰들을 이용할 수 있다. 본 논문에서는 MCS를 구현을 위하여 관련 자료(T.122/T.125)를 분석하였으며, 이를 근간으로하여 알고리즘을 개발하고, 이를 구현하였다.

2. MCS의 소개

MCS(Multipoint Communication Service)는 상호 대화적인 다양한 멀티미디어 응용들을 지원하기 위하여 설계되었으며, T.123에서 규정한 다양한 Network 상에서 다양한 여러 응용 객체들 사이에 양방향 나자간 통신을 제공한다. MCS는 일대일의 MCS연결들을 통하여 다지점 영역을 형성한다. 이 다지점 영역내에서 임의의 응용 객체는 영역내 다른 객체들에게 채널들을 이용하여 정보를 전달할 수 있으며, 자원 충돌 해결을 위하여 토큰들을 이용할 수도 있다.

MCS를 이용하는 MCS 사용자란 MCS provider가 제공하는 MCS 서비스를 요청하는 상위 응용들을 일컬으며 특정 MCS provider에 하나 이상의 MCS 사용자가 연결될 수 있다. 각 MCS 사용자들은 최상위 MCS provider로의 접속 서비스를 통해 다자간 통신을 위한 영역을 설정하고 최상위 MCS provider는 동일 영역내의 사용자들을 관리한다.

따라서 MCS 이용자는 먼저 그가 속한 MCS 제공자와 상대방의 MCS 제공자 사이에 MCS 연결을 함으로서 영역을 확장하며, 사용자들은 이 영역에 등록할 수 있다. 영역이 형성되면, MCS 사용자는 그가 필요로 하는 정보를 획득하기 위하여 적절한 채널들에 가입할 수 있다. 또한 사용자들에게 유용한 자원들을 관리하기 위하여 토큰들을 이용할 수 있다.

1. MCS에서의 채널 관리

영역 형성 및 사용자 접속이 끝난 후, 영역내 사용자들 사이에 다양한 형태로 정보를 교환하기 위해서는 적절한 채널들에 가입해야 한다. MCS내 채널들은 영역내 고유 번호를 가지고 있으며, 사용자는 Channel Join, Channel Leave를 이용하여 특정 채널에 가입 및 탈퇴를 할 수 있다. MCS내의 채널의 종류는 다중 전송 채널, 단일 전송 채널, 사적 채널이 있으며, 이를 설명하면 다음 과 같다.

다중 전송 채널(Multicast Channel)은 영역내 모든 사용자들 또는 일부 사용자들에게 정보를 전달하기 위하여 사용된다. 어떤 다중 전송 채널은 회의에 참가한 모든 참석자들이 가입할 수도 있으며, 어떤 다중 전송 채널을 참석자들의 일부가 가입할 수도 있다. 즉 사용자는 적절한 다중 전송 채널들에 가입함으로써, 가입한 채널들에 전송되는 정보를 습득할 수 있으며 가입하지 않은 원하지 않는 정보는 거절할 수 있다. 만일 사용자가 특정 다중 전송 채널에 정보를 전송만을 원할 경우 그 채널에 가입할 필요는 없다. 즉 채널에 가입하지 않고도 그 채널에 정보를 전송할 수 있다.

단일 전송 채널(Single Member Channel)은 영역에 접속된 각 사용자에게 할당되는 채널로 일반적으로 영역내 사용자 고유 번호로 이용된다. 이 채널은 각 사용자만이 가입이 허락되며, 영역내 임의의 구성원에게 정보를 전달하기 위해서는 그 구성원에 할당된 단일 전송 채널에 정보를 전달하면 된다.

다중 전송 채널은 다수의 사용자들 사이에 정보를 한번에 교환하는 장점을 가지고 있지만 영역내 임의의 구성원이 임의의 다중 전송 채널에 가입할 수 있기 때문에 영역내 임의의 사용자들 사이만 정보를 교환해야 할 때는 문제가 있다. 이를 위하여 사적 채널(Private Channel)이 있으며, 이 채널은 다중 전송 채널의 특성을 가지고 있으나 다른 점은 모든 사용자가 가입되는 것이 아니라, 초청된 사용자들만이 가입할 수 있는 특성을 가지고 있다. 이를 위하여 채널을 관리하는 관리자가 필요하며, 이는 이 채널을 요청하고 할당받은 사용자가 관리자가 된다. 즉 사용자는 Channel Convene를 통하여 새로운 사적 채널을 할당받고 그 채널의 관리자가 된다. 이 관리자는 Channel Admit을 통하여 특정 사용자를 이 채널에 초대한다. 또한 현재 초대되었거나 가입된 사용자를 강제로 추방하기 위하여 Channel Expel을 사용한다. 또 이 채널의 관리를 포기하고 반납하기 위하여 Channel Disband를 사용한다. 이 채널에 정보를 전달하기 위해서는 사용자는 이 채널에 초청되어 있어야 하며, 초대된 상태에서 이 채널에 가입한지 않는 경우, 이 채널

의 정보를 수신하지 않고 전송만 할 수 있다.

각 채널 id는 정적 채널 id와 동적 채널 id로 구분되며, 정적 채널 id는 영역이 존재할 때까지 같이 존재하며 각 사용자들은 언제든지 가입할 수 있다. 동적 채널 id는 다시 사용자 id, 사적 채널 id, 지정 채널 id로 구분되며 각각을 설명하면 다음과 같다. 사용자 id는 MCS 사용자 id로 지정된 사용자만이 가입할 수 있으며, 사적 채널 id는 이 채널 관리자와 이 관리자가 초대한 사용자들만이 가입할 수 있다. 또한 지정 채널 id는 채널 id가 0인 채널에 가입시 만들어지며 이때 새로이 주어진 채널 id를 할당받는다. 그 뒤 모든 사용자가 이 채널에 가입할 수 있으며, 가입된 모든 사용자가 탈퇴하면 이 채널은 제거된다.

정적 채널 id는 1에서 1000까지 할당되며, 동적 채널 id는 1001에서 65535까지 할당된다. 동시에 운용 가능한 채널의 수는 MCS 영역 생성시 parameter에 의하여 결정된다. 정적 채널 id는 임의의 사용자들이 가입시 사용중이 되며, 동적 채널 id는 채널이 생성되면 사용중이 된다. 만일 사용중인 채널의 수가 영역이 운용할 수 있는 채널의 수보다 많아지면 Channel Join, Channel Convene, Attach User의 MCS primitive들을 사용시 실패한다.

2. MCS에서의 데이타 전송

다수의 MCS Provider들이 연결되고 같은 영역을 형성한 뒤, 임의의 사용자들이 이 영역에 접속한 후, 적절한 형태의 채널에 가입하면, 사용자는 다양한 형태로 정보를 교환할 준비가 되었다고 말할 수 있다. MCS에서 데이타 전송을 위하여 Send Data와 Uniformly Sequenced Send Data를 제공한다.

Send Data 서비스는 특수한 형태로 일대일을 포함하는 일대다 형태의 정보 전송을 제공한다. 상대방의 사용자 id에 해당하는 채널에 데이타를 전송하는 것은 일대일 정보전송에 해당되며, 사용자가 다수 속해있는 채널에 데이타를 전송하는 것은 일대다 정보전송에 해당된다. 또 이를 이용하여 다대일 다대다 정보전송도 가능하다. Simple Send 정보전송은 채널로 전송한 데이타가 최단경로를 통하여 채널내 사용자에게 전송된다. 예를 들어, 동일 채널내에 임의의 사용자가 동일 MCS Provider안에 있다면 전송된 정보는 Provider내에서 직접 전달된다. 이 방법은 뒤에 설명할 Uniformly Sequenced Send Data와 구분된다.

Uniformly Sequences Send Data는 Simple Send와는 두 가지 점이 다르다. 첫째는, 특정 채널로 전송된 데이타는 일단 Top Provider에게 전송한 뒤, Top Provider가 이 채널에 속한 사용자들에게 이 데이타를 전송한다. 둘째는, 채널내 모든 사용자에게 정보가 전송된다는 점이다. 만일 전송자가 이 채널에 가입한 상태에서 정보를 전달하였다면, 이 전송자에게도 자신이 전송한 정보가 전달된다. 이는 다대다 전송에서 모든 사용자들이 동일한 순서로 정보를 수신해야하는 경우 유용한 전송방식이다.

3. MCS에서의 토큰 관리

토큰은 상호 배타적인 접근을 구현하기 위한 수단을 제공한다. 예를 들어, 주어진 시간에 주어진 자원을 한 사용자만이 소유해야 하는 다지점 응용에서, 토큰은 각 자원과 연관되어 이를 보장할 수 있다. 즉 어떤 사용자가 특정 자원을 사용하길 원한다면, 그 사용자는 대응되는 토큰을 요청해야 하며, 만일 아무도 그 토큰을 소유하지 않았으면 이를 소유할 수 있다. 즉 토큰을 소유한 사용자만이 그 자원을 이용할 수 있으며, 다른 사용자들도 그 자원을 사용하게 하기 위하여 그 자원을 다 사용한 사용자는 해당 토큰을 반납해야 한다. 이러한 토큰을 획득하기 위해서는 Token Grab 또는 Token Inhibit 서비스를 이용하여야 하며, 각각을 설명하면 다음과 같다.

Token Grab 서비스는 단지 하나의 사용자만이 주어진 토큰을 배타적으로 소유하는 것을 허락한다. 만일 주어진 토큰을 어떤 사용자가 사용하고자 할때, 그 사용자는 Token Test를 통하여 그 토큰의 현재 상태를 파악한다. 만일 그 토큰을 다른 사용자가 소유하고 있다면 Token Please를 이용하여 토큰이 필요함을 그 토큰을 소유한 사용자에게 알린다. 토큰을 소유하고 있는 사용자는 Token Give를 이용하여 특정 사용자에게 토큰의 소유권을 이양할 수 있다. 또한 Token Release를 이용하여 해당 토큰을 반납할 수도 있다.

Token Inhibit 서비스는 다수의 사용자가 토큰을 이용하는 것을 허락한다. 이를 이용하여 다수의 사용자는 여러 유용한 일을 할 수 있다. 예를 들어, 만일 어떤 일을 다수의 사용자들이 시작한 후 모든 사용자

가 이일을 끝낸 후 다음일을 해야 하는 상황이 존재할 때, 각 사용자는 일을 시작하기전 특정 토큰을 정한 후 Token Inhibit을 이용하여 토큰을 사용하고 일을 끝낸 후 Token Release를 이용하여 토큰을 반납한다. 이를 이용하면 모든 이용자는 Token Test를 이용하여 이 토큰이 사용중이면 이용자들이 모두 일을 끝낸 상태가 아니며 만일 이 토큰이 사용중이 아니면 모든 사용자가 주어진 일을 끝냈음을 알 수 있다.

토큰 id는 1부터 65535까지 할당되며, 이 범위 안의 모든 숫자는 유효한 토큰 id가 된다. 한번에 사용가능한 토큰의 수는 영역의 생성시 결정된다. 토큰이 grabbed 되었거나 inhibit 되었을 때 이는 사용중인 토큰이 된다. 만일 사용중인 토큰의 수가 제한된 토큰의 수를 초과하면 Token Grab, Token Inhibit의 MCS 요소들이 실패한다.

4. MCS 서비스 요소들

MCS가 제공하는 서비스 요소들을 나열하면 다음과 같다.

(1). MCS 영역 관리 요소들
- MCS Connect Provider
- MCS Disconnect Provider
- MCS Attach User
- MCS Detach User

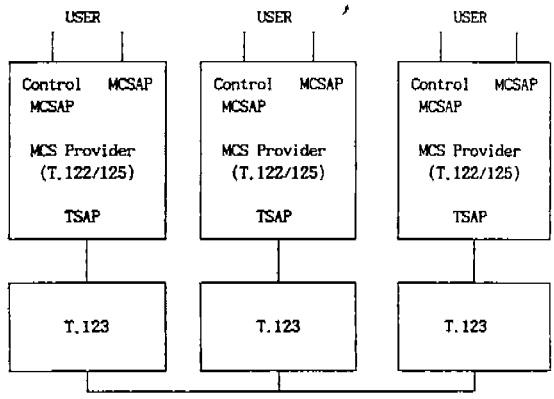
(2). MCS 채널 관리 요소들
- MCS Channel Join
- MCS Channel Leave
- MCS Channel Convene
- MCS Channel Disband
- MCS Channel Admit
- MCS Channel Expel

(3). MCS 정보 전송 요소들
- MCS Send Data
- MCS Uniform Send Data

(4). MCS 토큰 관리 요소들
- MCS Token Grab
- MCS Token Inhibit
- MCS Token Give
- MCS Token Please
- MCS Token Release
- MCS Token Test

5. ITU-T T.125

T.125는 MCS 프로토콜 규격으로, MCS provider들 사이의 데이타 및 제어 정보의 교환을 위한 규약을 위한 과정들을 소개하고, 데이타 및 제어정보의 교환에 사용되는 MCS PDU(Protocol Data Unit)들의 구조 및 부호화를 정의하고 있다. 즉, T.125는 사용자가 MCS를 실제적으로 이용하기 위한 MCS 요소들을 소개하고 있으며, T.125는 T.122에서 소개한 MCS 요소들을 실제적으로 구현하기 위한 방안을 소개하고 있다. T.122, T.125, T.123의 관계는 그림 1과 같이 도시할 수 있다[5][6][7].



(그림 1) Model of T.122/125 MCS Layer

3. MCS 구현을 위한 알고리즘의 소개

3.1 전체적인 알고리즘의 구성도

MCS 알고리즘은 제어부(Control process), 영역부(Domain process), 연결부(Attachment process), 종단부(Endpoint process)의 네개의 Process로 구분된다. 제어부(Control process)는 MCS 노드 관리자에 의하여 생성되며, MCS를 전체적으로 관장하는 역할을 한다. 즉 노드 관리자의 요청에 의하여 또는 MCS내부의 사정에 의하여, 제어부는 영역부, 연결부, 종단부

를 생성 또는 소멸시킨다. 제어부는 MCS의 존재와 더불어 생성되며, 각 MCS Provider당 한 개의 제어부만이 존재한다. 영역부는 MCS 알고리즘중 가장 핵심적인 역할을 하며, 하나의 영역당 하나의 영역부가 존재한다. 영역부의 역할은 MCS PDU(Protocol Data Unit)를 입력으로 받아 이를 처리하는 역할을 한다. 연결부는 AUrq(Attach User request) MCS 요소에 의하여 생성되며, 사용자 프로그램의 요청에 의하여 생성된다. 일반적으로 사용자 프로그램은 하나의 영역당 한 개의 창구를 이용하기 때문에, 한 개의 연결부를 가진다. 따라서, 다수의 영역에 접속할 경우 각 응용 프로그램당 다수의 연결부를 가질 수 있다. 연결부는 사용자 프로그램으로부터 MCS 요소들을 받아 이를 MCS pdu로 바꾼 후 이를 영역부로 전달하며, 또한 영역부로부터 MCS pdu를 받아 이를 사용자 프로그램에 전달하는 역할을 한다. 종단부는 영역 설정시 생성되며, MCS내 다수의 영역이 설정되는 경우 다수의 종단부가 설정된다. MCU(Multipoint Control Unit)의 경우, 하나의 영역당 다수의 MCS Provider들이 연결되기 때문에 다수의 종단부들이 생성된다. 종단부의 역할은 MCS Provider사이에 MCS pdu를 교환하는 창구 역할을 한다.

3.2 영역부의 알고리즘 소개

영역부는 MCS 요소의 요청을 받아 이를 처리하는 MCS에서 가장 핵심적인 부분이며, 제어부로부터 생성되고 소멸된다. 또한 제어부에 의하여 생성된 연결부 및 종단부와 형성된 통로를 할당받으며, 연결부 및 종단부로 부터 전달된 MCS PDU를 처리한다. 이 과정을 설명하면 다음과 같다. 먼저, MCS PDU의 형태가 유효한 형태가 아니면 이를 무시한다. 만약 유효하다면, 이 MCS PDU를 처리하는 과정이 MCS가 MCU의 역할을 할 때와 그렇지 않을 때로 구분된다. 만약, MCU라면, 먼저 Top Provider 과정을 수행한 후, Apply PDU 과정을 수행한다. 그렇지 않을 경우, Apply PDU 과정만을 수행한다. MCU로 이용되지 않을 경우 Apply PDU 과정만을 수행한다. 즉, MCU의 경우, MCS PDU안에 포함된 내용이 먼저 Top Provider 과정에서 처리된 후, 그 내용이 변경된다. 그 뒤 이 내용이 Apply PDU에 의하여 최종적으로 변경된다. 이와 관련한 알고리즘은 아래와 같다.

```
Domain Procedure
{
    Initialize Resources;
    while(TRUE){
        Receive the pdu from another process
        If the pdu is received from Control process,
            the corresponding procedure is executed.
        If the pdu is received from Attachment process
                and Endpoint process,
            Get the corresponding portal
                of the process to send the pud, and
            Process the pdu.
        }
}
Process the pdu
{
        Validate the MCSPDU and perform other checks,
            depending on its type.
        If this provider is MCU provider
            Execute the Top Provider routine.
        Apply PDU routine.
        If Result is error, then
            Send Drop portal pdu to control process, and
            send RJum pdu to the corresponding process.
}
Top Provider
{
    MCU Processing:
    The buffer containing an MCSPDU will be processed next
    by Apply_PDU. Its contents may first be modified here.
}
Apply PDU
{
    This is a large case selection based on MCSPDU kind.
    These actions and updates to the information base
```

take place in both the top and subordinate MCS providers

}

3.3 제어부의 알고리즘 소개

제어부는 MCS 내 한 개만 존재하며, MCS 초기화 과정에서 생성된다. 제어부의 역할은 첫째, MCS 노드 관리자 및 응용 프로그램의 요청에 의하여 연결부, 영역부, 종단부를 생성 또는 소멸시키며, 이들과 정보를 교환한다. 둘째, 연결부를 생성하고, 영역부와 연결부 사이에 통로를 설정하고, 이를 영역부에게 알린다. 셋째, 종단부를 생성하고, 영역부와 종단부사이에 통로를 설정하고, 이를 영역부에게 알린다. 넷째, 노드 관리자의 요청에 의하여 임의의 영역에 MCS를 연결하고, 분리한다. 이와 관련한 알고리즘은 아래와 같다.

Control Process

{

while(TRUE){

1. If this is MCU process and
   Receive the TC(T. Connect. Indication) pdu
   from transport.
   Then, Allocate portal, and
   create an endpoint process

2. Receive the pdu from the corresponding process.

3. If the pdu (CPrq, CPrs, DPrq, CD, DD, EXIT) is
   received from Node Controller,
   the corresponding procedure is executed.

4. If the kind of the pdu is
   AUrq(Attach User Request),
   Create the Attachment process, and
   Send the ATTACHED pdu to Domain process.

5. If the kind of the pdu CR(Connect Response),
   Then Send the pdu to Node controller.

6. If the kind of the pdu CI(Connect Initial),
   Then Send the CPin(Connect Provider Indication)
   pdu to Node controller.

7. If the pdu (SP, DP, RP) is received
   from Domain process,
   Execute the corresponding procedure.

}

}

3.4 연결부의 알고리즘 소개

연결부는 하나의 응용 프로그램당 한 개씩 존재하며, 제어부에 의하여 생성되고, 소멸된다. 그 역할로는, 응용프로그램에 의하여 전달받은 MCS PDU를 영역부에 전달하며, 영역부에 의하여 전달받은 MCS PDU를 응용 프로그램에 전달한다. 이와 관련한 알고리즘은 아래와 같다.

Attachment Process

{

Send the AUrq(Attach User Request) pdu to Domain process

while(TRUE){

Receive the pdu from another process.
If the kind of pdu is one received
from User application
then, the corresponding procedure is performed and
the pdu is send to Domain process.

If the kind of pdu is one received
from Domain process
then, the corresponding procedure is performed and
the pdu is send to User application.

If the kind of pdu is EXIT
then, the attachment process is ended.

}

}

3.5 종단부의 알고리즘 소개

종단부는 MCS 영역 연결시 제어부에 의하여 생성되며, MCS 영역 분리시 제어부에 의하여 소멸된다. 그 역할로는, 첫째, 임의의 영역에 MCS를 연결할 때, 제어부로부터 Connect MCS pdu를 전달받아 Transport로 전달한다. 또한, Transport로부터의 Connect

MCS PDU를 전달받아 제어부로 전달한다. 둘째, 연결이 설정된 후, 다른 MCS로부터, MCS PDU를 전달받아 영역부에 전달한다. 셋째, 영역부로부터 MCS PDU를 전달받아 Transport로 전달한다. 이와 관련한 알고리즘은 아래와 같다.

```
Endpoint Process
{
    If this is MCS endpoint process,
        Send the TCrq(T_Connect_Request) pdu
            to Control process in MCU.
        Receive the pdu from Transport,
        and process the pdu.
    Else if this is MCU endpoint process,
        Send the TCrs(T_Connect_Response) pdu
            to Transport.
    Connect_PDU_ready is TRUE;
    while(Connect_PDU_ready){
            Receive the pdu from another process.
            If the pdu is received from Control process
              then, if the kind of pdu is CR,
                 Connect_PDU_ready is FALSE
              Send the pdu to Transport.
            If the pdu is received from Transport,
              then, if the kind of pdu is CR
                 Connect_PDU_ready is FALSE;
              Send the pdu to Control process.
    }
    while(TRUE){
        Receive the pdu from the corresponding process.
        If the pdu is received from Domain process,
          Send the pdu is send to Transport.
        If the pdu is received from Transport,
            then, the corresponding procedure is executed.
        If the pdu is received from Control process
            then, the corresponding procedure is executed.
    }
}
```

4. MCS 구현 환경

MCS의 구현은 unix 운영체제를 이용하는 컴퓨터 및 windows/NT를 이용하는 컴퓨터 위에서 구축되었으며 실험은 unix 운영체제를 이용하는 SUN computer와 windows/NT 운영체제를 이용하는 IBM PC를 이용하였다. 통신 환경은 Ethernet을 이용하는 LAN 을 이용하였으며, ATM card를 이용하는 ATM 망을 이용하여 실험하고 있다.

실험 환경은 LAN으로 연결된 3대의 SUN workstation과 2대의 IBM-PC를 이용하였으며, 그중 1대의 SUN은 MCU로서 최상위 MCS 제공자 역할을 하며 모든 영역들과 채널 토큰들을 관장한다. 또한 나머지 2대의 SUN workstation 및 2대의 IBM-PC는 MCS 제공자를 두어 사용자의 요구에 의하여 최상위 MCS 제공자로의 연결 및 지역적으로 채널 및 토큰을 관리하고 있다. 또한 사용자 및 응용 프로그램의 MCS의 원활한 사용을 위하여 API를 정의하여 실험시 이를 이용하여 MCS를 실험하였다. 알고리즘의 구현은 C 언어를 이용하여 구현되었으며, MCS 간의 통신을 위한 T.123 protocol stack으로는 TCP/IP를 이용하였다. 알고리즘에서 제어부, 영역부, 종단부, 연결부는 각각의 프로세서로 구현하였으며, 따라서 하나의 MCS는 작동시 최소한 4개의 프로세서가 수행된다. 프로세서간의 통신은 메시지 교환 방식을 이용하였으며, 수행 속도의 개선을 위하여 현재 공유 메모리형을 이용하는 방식도 구현 중이다.

5. 결 론

현재 및 미래의 다양한 통신망에서 멀티미디어 회의 관련 응용 서비스를 구현하기 위해서는 기본적인 회의 서비스에 관련된 규약이 필요하며 이와 관련하여 제정된 규약이 MCS(Multipoint Communication Ser-

vice)이다. MCS는 상호 대화적인 다양한 멀티미디어 응용들을 지원하기 위하여 설계되었으며, T.123에서 규정한 다양한 Network상에서 다양한 여러 응용 객체들 사이에 양방향 다자간 통신을 제공한다. 또한 T.125는 MCS 프로토콜 규격으로, MCS provider들 사이의 데이타 및 제어 정보의 교환을 위한 규약을 위한 과정들을 소개하고, 데이타 및 제어정보의 교환에 사용되는 MCS PDU(Protocol Data Unit)들의 구조 및 부호화를 정의하고 있다. 즉, T.122는 사용자가 MCS를 실제적으로 이용하기 위한 MCS Primitive들을 소개하고 있으며, T.125는 T.122에서 소개한 MCS Primitive들을 실제적으로 구현하기 위한 방안을 소개하고 있다. MCS는 일대일의 MCS연결들을 통하여 다지점 영역을 형성한다. 이 다지점 영역내에서 임의의 응용 객체는 영역내 다른 객체들에게 채널들을 이용하여 정보를 전달할 수 있으며, 자원 충돌 해결을 위하여 토큰들을 이용할 수 있다. 본 논문에서는 MCS를 구현을 위하여 관련 자료(T.122/T.125)를 분석하였으며, 이를 근간으로 하여 알고리즘을 개발하고, 이를 구현하였다.

참 고 문 헌

[1] K. Watabe, S. Sakata, K. Maeno, H. Fukuoka, and T. Ohmori, "Distributed Desktop Conferencing System with Multiuser Multimedia Interface," IEEE journal on Selected Areas in Comm. Vol. 9. pp. 531-539, 1991.

[2] T. Ohmori, K. Maeno, S. Sakata, H. Fukuoka, and K. Watabe, "Distributed Cooperative Control for Application Sharing Based on Multiparty and Multimedia Desktop Conferencing System: MERMAID," Multimedia Communication. pp. 112-131, 1992.

[3] W. J. Clark, "Multipoint Multimedia Conferencing," IEEE Communication Magazine, May pp. 44-50, 1992.

[4] S. Masaki, T. Arikawa, H. Ichihara, M. Tanbara, and Shinamura, "A Promising Groupware System for B-ISDN: PMTC," Multimedia Communication --pp. 190-198, 1992.

[5] ITU-T Recommendation T.122, "Multipoint Communication Service for Audiographics and Audiovisual Conferencing Service Definition", 1993.

[6] ITU-T Recommendation T.125, "Multipoint Communication Service Protocol Specification", 1994.

[7] ITU-T Recommendation T.123, "Protocol Stacks for Audiographic and Audiovisual Teleconference Applications", 1994.

성 동 수

1987년 한양대학교 공과대학 전자공학과(학사)
1989년 한국과학기술원 전기 및 전자공학과(석사)
1992년 한국과학기술원 전기 및 전자공학과(박사)
1993년 한국과학기술원 정보전자연구소 연구원
1993년~현재 경기대학교 공과대학 전자공학과 조교수
관심분야: 멀티미디어 통신, 병렬처리컴퓨터, 지능컴퓨터.

현 동 환

1989년 광운대학교 전자계산기공학과(학사)
1991년 광운대학교 대학원 전자계산기공학과(공학석사)
1991년~1997년 한국전자통신연구원 선임연구원
1997년~현재 영진전문대학 사무자동화학과 전임강사
관심분야: 멀티미디어 통신, 멀티미디어 응용서비스

함 진 호

1982년 한양대학교 공과대학 전자공학과(학사)
1984년 한양대학교 공과대학 전자통신공학과(석사)
1994년~현재 한국전자통신연구원 정보통신표준연구센터 멀티미디어표준 연구실 실장
한양대학교 공과대학 전자통신공학과(박사과정)
관심분야: 멀티미디어 통신, 정보통신.